Facultad de Ingeniería

Trabajo de Investigación

“Aplicaciones de software que permitan la integración con otras aplicaciones de otras plataformas en la empresa A”

Autores:

BARRERA GARAY, David Ian – 1525393

POMA CHAVEZ, Edgar Lenon– 1622174

Para obtener el Grado de Bachiller de

Ingeniería Software

Lima, julio 2020

Dedicatoria

Dedicado para nuestros padres, que con la entrega de su trabajo pudieron hacer posible, nuestros estudios, además de ofrecer su apoyo moral en todo momento, para ayudarnos a continuar con este proceso de aprendizaje y maduración. También a aquellos compañeros y amigos que nos apoyaron y nos juntaron a lo largo de diversos cursos, haciendo de estos más sencillo con su apoyo.

Agradecimiento

Agradecemos a aquellos docentes que confiando en nosotros nos evaluaron de forma meticulosa, dando su esfuerzo para proporcionarnos un mejor entendimiento de nuestro trabajo de investigación; por la supervisión y empeño que demostraron en la evaluación de nuestros avances. Por ello, nos motivamos también a dar el mejor esfuerzo y empeño, para poder honrar aquello que hicieron por nosotros.

ÍNDICE DE CONTENIDO

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

CAPÍTULO I: RESUMEN

1.1. TÍTULO

Aplicaciones de software que permitan la integración con otras aplicaciones de otras plataformas en la empresa A

1.2. RESUMEN

Las empresas usualmente poseen diversos sistemas informáticos (SI), originado por la necesidad de su labor cotidiana e influenciada por la demanda de sus clientes y la competitividad actual que es generalmente agresiva. Ocurre en más de una organización que aquellos SI se desenvuelven muchas veces de forma independiente y aislada, es decir sin conexión entre ellas. Por ello, pueden aparecer constantemente operaciones redundantes, con baja o nula optimización dentro de la empresa. Pero estas operaciones podrían agilizarse, con la integración de dichos SI a través de la utilización de una aplicación.

Ante esta situación, en primera instancia se procede a desarrollar la presente investigación de revisión bibliográfica donde se pretende encontrar suficiente información que respalde y permita el desarrollo de una Aplicación Híbrida capaz de lograr dicha integración.

La importancia de la presente Investigación de revisión bibliográfica radica en que permite escoger información disponible de una diversidad de fuentes que fueron: Repositorios de las universidades nacionales e internacionales, SCOPUS y Google Académico, entre otros. Es decir, pretende reunir literatura científica con respecto a la “Aplicaciones de software que permitan la integración con otras aplicaciones de otras plataformas en la empresa”

Aunque, la principal limitación es el problema económico- social desatado por el COVID-19. Se pudo superar, haciendo uso de los medios digitales.

Finalmente, la información analizada puede formar parte de una base científica confiable, ya que permite lograr el cumplimiento de los objetivos tanto la general como las específicas y responde nuestra pregunta de investigación.

1.3. PALABRAS CLAVE

Integraciones, Servicios WEB, REST, JSON, Token.

CAPÍTULO II: INTRODUCCIÓN

2.1. CONTEXTO

Las empresas presentan grandes dificultades para lograr adaptarse a los cambios vertiginosos que demanda el mercado, haciendo que estas enfrenten constantemente importantes desafíos, a los cuales en su mayoría no saben responder. Haciendo que las diferentes organizaciones empresariales, ingresen en un círculo que solo avizora dificultades en el intercambio de información. Además, en la actualidad se está viviendo una situación sin igual, el cual es la pandemia de “Covid 19”, este ha perjudicado económicamente a un sinfín de empresas, haciendo aún más difícil el que se adapten tecnológicamente.

La creación de Sistemas Informáticos no integradas aumenta el esfuerzo en desmedro de los resultados, aumentan el tiempo utilizado en proyectos y actividades diarias, así como los costos y gastos, con casi nula ventaja en su

competitividad. Por un lado, estas dificultades ocurren principalmente debido a que el sector tecnológico está evolucionando a gran velocidad y constantemente salen al mercado nuevas plataformas, equipamientos, sistemas de almacenamiento, software avanzado y nuevas formas de procesamiento de datos, elementos importantes para procesar la gran cantidad de datos existentes y que no tiene comparación con las de hace algunas décadas atrás. Por otro lado, las empresas, por su afán de cumplir con los proyectos asignados van, creando diferentes Sistemas Informáticos, según las áreas que lo necesiten, esto origina desorden, debido a la posesión de diferentes bases de datos descentralizados, ajenos entre sí, funcionando para su respectivo sistema, sin cooperación conjunta y sin integración.

Además, una empresa con SI que no se encuentran integrados, pierde tiempo y genera dificultades para la gestión de aplicaciones e información en desmedro constante del valor agregado y de los procesos que se ejecutan. Por ello, para el desarrollo de las empresas, la integración de los SI conforma un punto clave.

Solo las empresas que cambian sus procesos en el manejo de la información implementan nuevas tecnologías, la integran y aprovechan, podrán lograr alcanzar mayores éxitos.

Finalmente, como resultado de esta investigación bibliográfica, se puede indicar que la información proporcionada por los diferentes documentos recopilados, permiten el planteamiento de una aplicación híbrida con APIs REST que integra aplicaciones informáticas de otras plataformas de una empresa, con ello se cumplen los objetivos y se responden las preguntas de esta investigación.

2.2 OBJETIVOS DE LA INVESTIGACIÓN

2.2.1 OBJETIVO DE GENERAL

Proporcionar información que permita el planteamiento de una aplicación híbrida con APIs REST para la integración de aplicaciones informáticas de otras plataformas de una empresa privada.

2.2.2 Objetivos Específicos

a. Identificar las funcionalidades que debe tener la aplicación híbrida.
b. Identificar la tecnología para la comunicación entre la aplicación hibrida con las diferentes aplicaciones informáticas de la empresa.
c. Describir una aplicación híbrida que permita la gestión de datos de la empresa.
d. Describir que tecnología permite establecer una comunicación segura de datos

2.3 PREGUNTAS DE INVESTIGACIÓN

2.3.1 Pregunta General

¿La información proporcionada permite el planteamiento de una Aplicación Híbrida con APIs REST que integra aplicaciones informáticas de otras plataformas de una empresa?

2.3.2 Preguntas Específicas

a. ¿Cuáles son las funcionalidades que debe tener la aplicación híbrida?

b. ¿Cómo se da la comunicación entre la aplicación hibrida con las diferentes aplicaciones informáticas de la empresa?
c. ¿Cómo una aplicación híbrida permite la gestión de datos de la empresa?
d. ¿Cuál es la tecnología que permite establecer una comunicación segura de datos?

2.4 ESTRUCTURA DE LA INVESTIGACIÓN

La investigación presente, se hizo del siguiente modo cuenta con los siguientes procedimientos, para el desarrollo y elaboración del tema elegido.

Primero, en base al Tema y palabras claves otorgadas por la Institución, se decide iniciar una investigación bibliográfica, con el fin de recolectar información confiable que permita cumplir objetivo y responder preguntas. Luego, se plantearon los objetivos que permitan responder y justificar las preguntas de investigación.

Como método, se procedió a reunir e investigar diversas fuentes científicas y académicas que guarden relación a la implementación con el tema y sus tecnologías. Se indaga en trabajos de investigación y proyectos de la base de datos SCOOPUS, Repositorios de Universidades Nacionales e Internacionales.

A continuación, con las fuentes reunidas, se procedió a clasificar según su enfoque, en: Seguridad de Datos, APIS REST, Servicios WEB, Aplicación híbrida, Integración de aplicaciones. Realizando en cada una, análisis del contexto y tecnologías usadas, concluyendo con comentarios que recalca cómo podría aportar a nuestra investigación.

También, con dichas fuentes, se usarán elementos paratextuales, para poder encontrar puntos de desencuentro y similitudes entre las propuestas de las

fuentes. Estos elementos son: Cuadros, donde se expone que combinación de tecnologías usa dicha fuente, para encontrar diferencias y similitudes en el uso de estas.

Finalmente, con la información analizada, procedemos a confirmar, si se cumplen los objetivos y con ello si las preguntas obtienen una respuesta lógica, que en conjunto permitan establecer que es factible plantear una solución referente al tema de esta investigación: “Aplicaciones de software que permitan la integración con otras aplicaciones de otras plataformas en la empresa A”

CAPÍTULO III: METODOLOGIA

3.1 FUENTES DE INFORMACIÓN

Para la realización de investigaciones de revisión bibliográfica, es necesaria la recolección de información. Según Parraguez, Chunga, Flores, & Romero (2017), la recolección de información es importante para el inicio de una investigación, en este proceso deben obtenerse información pertinente sobre libros, trabajos de investigación como tesis, artículos científicos, etc.

Las TIC, son herramientas valiosas en el trabajo universitario, ya que brindan la posibilidad de optimizar la investigación y los procesos de estudio. Además, permite llevar a cabo un trabajo importante, individual, emprendedor, colaborativo e independiente.

Una Revisión Bibliográfica para el autor Bernardo (2010), es un texto diseñado para brindar un resumen completo referente a las lecturas que aportan elementos sobre el tema en cuestión a investigar para luego sacar conclusiones o discusiones.

Por ello, el presente estudio es una Investigación de Revisión Bibliográfica porque presenta de manera ordenada los resultados obtenidos en diversas investigaciones previas relacionadas a la Implementación de aplicaciones de software que permitan la integración con otras aplicaciones de otras plataformas.

Los recursos elegidos para este trabajo de investigación se deben a que son bases de datos que contienen, trabajos de investigación, tesis, artículos científicos, entre otros, referidos al tema y rubro, por ello nuestras fuentes de información son: Repositorios de la universidades nacionales e internacionales, SCOPUS y Google Académico, entre otros.

3.2 CRITERIOS DE BÚSQUEDA Y SELECCIÓN DE INFORMACIÓN

Los criterios de Búsqueda son las palabras o variables que permiten recabar información sobre anteriores investigaciones relacionadas con el tema de estudio y rubro de la investigación realizada. Se procedió principalmente realizar una búsqueda de documentos para ello se identificaron los siguientes términos de búsqueda: Integraciones, Servicios WEB, REST, JSON, Token. Además, con el fin de poder filtrar los artículos y obtener mejores resultados se utilizaron diferentes términos los cuales fueron:

Tabla 1: Términos de búsqueda utilizados

Términos de búsqueda utilizados

Integraciones con servicios web
Integraciones con API REST
Integración de aplicaciones y JWT
Integración de aplicaciones con Token
API REST y JSON
API REST con Token
Integración de sistemas informáticos
Aplicaciones híbridas con JWT
Implementación de JSON Web Token
Desarrollo de aplicación híbrida
Integración de sistemas con aplicación híbrida
API REST y JSON Web Token

Fuente: Elaboración propia

3.2.1 Criterios de inclusión

Para seleccionar la información, se aplicaron diferentes criterios de inclusión. El primer criterio se basa en los artículos originales publicados en la base de datos científica y repositorios académicos indexados en español de 2015 a 2020, esto ya

que se necesita información actual. Además, que las tecnologías usadas en estas no deben estar desfasadas. Luego como segundo criterio de inclusión se toman trabajos de investigación que hacen referencia al tema de la implementación de aplicaciones de software que permitan la integración con otras aplicaciones de otras plataformas en la empresa. Asimismo, se tomará en cuenta las investigaciones que tengan una temática similar (implementar software para gestión de almacenes, ventas, entre otros). Como tercer criterio de inclusión se seleccionan trabajos contenga como palabras clave las siguientes: integraciones, servicios web, REST, JSON, Token. Esto se realiza para que las investigaciones guarden relación con el tema abordado.

3.2.2 Criterios de exclusión

Se tomará en cuenta diferentes criterios de exclusión que ayudarán a que la información encontrada sea de calidad y contribuya al cumplimiento de los objetivos. En primer lugar, se encuentra el idioma de desarrollo de los trabajos de investigación, por lo cual se excluye a las investigaciones que no se encuentren en español o inglés. En segundo lugar, se tomará en cuenta el precio por el acceso al documento de investigación, por lo que si una investigación no es de acceso gratuito no se tomará en cuenta para nuestra investigación. Finalmente, se tomará en cuenta la carrera profesional de la investigación a tomar, por lo que todas las carreras a excepción de ingeniería de informática, sistemas, software, ciencias de la computación y afines no se tomarán en cuenta.

CAPÍTULO IV: DESARROLLO Y RESULTADOS

4.1 ESTRUCTURA DE LA INFORMACIÓN

En esta etapa de desarrollo brindamos información importante sobre las investigaciones, que hemos recopilado y analizado. Presentamos diferentes esquemas que brindaremos a continuación.

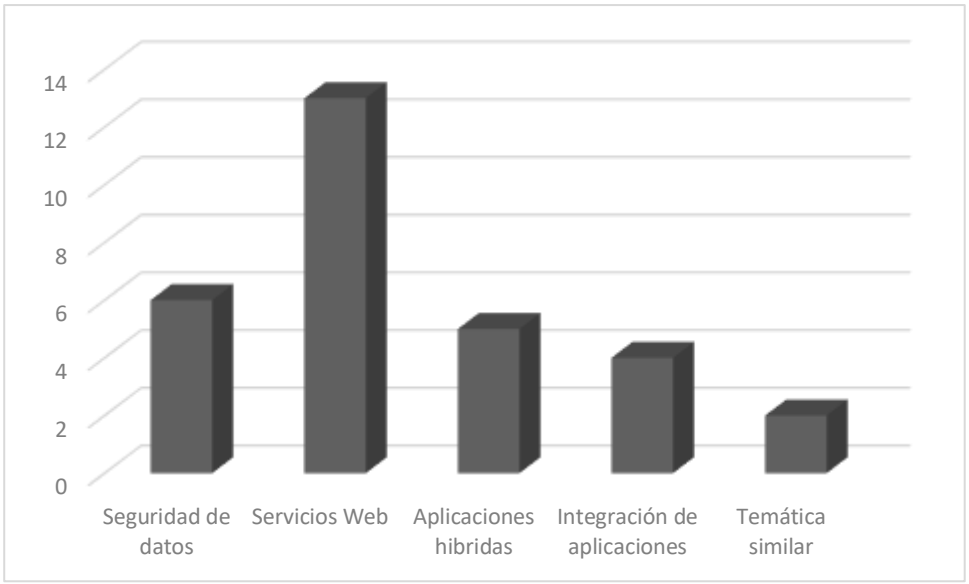


Figura 1: Investigación por temática

Fuente: Elaboración propia

En este primer esquema podemos observar que la cantidad de investigaciones revisadas para poder realizar esta revisión bibliográfica se visualiza que el punto que más información se ha encontrado es sobre “servicios web” aunque igualmente estos proyectos también abordan los demás puntos clave como lo son “seguridad de datos” he “integración de aplicaciones”. Asimismo, se observa que el punto “temática similar” es la que tiene menos fuentes investigadas, esto se debe a que en este punto nos centramos en encontrar información similar sobre una situación específica para el uso de una aplicación que permita la integración de otros softwares. Para nuestro caso seleccionamos la temática “Integración de aplicaciones”.

Figura 2: Investigaciones por países

Fuente: Elaboración propia

En este esquema se observa el país de origen de las diferentes fuentes académicas recopiladas. Los trabajos recopilados tienen como protagonista a los de origen peruano y española, con un 13% para cada uno respectivamente. Seguidas por India con un 10%.

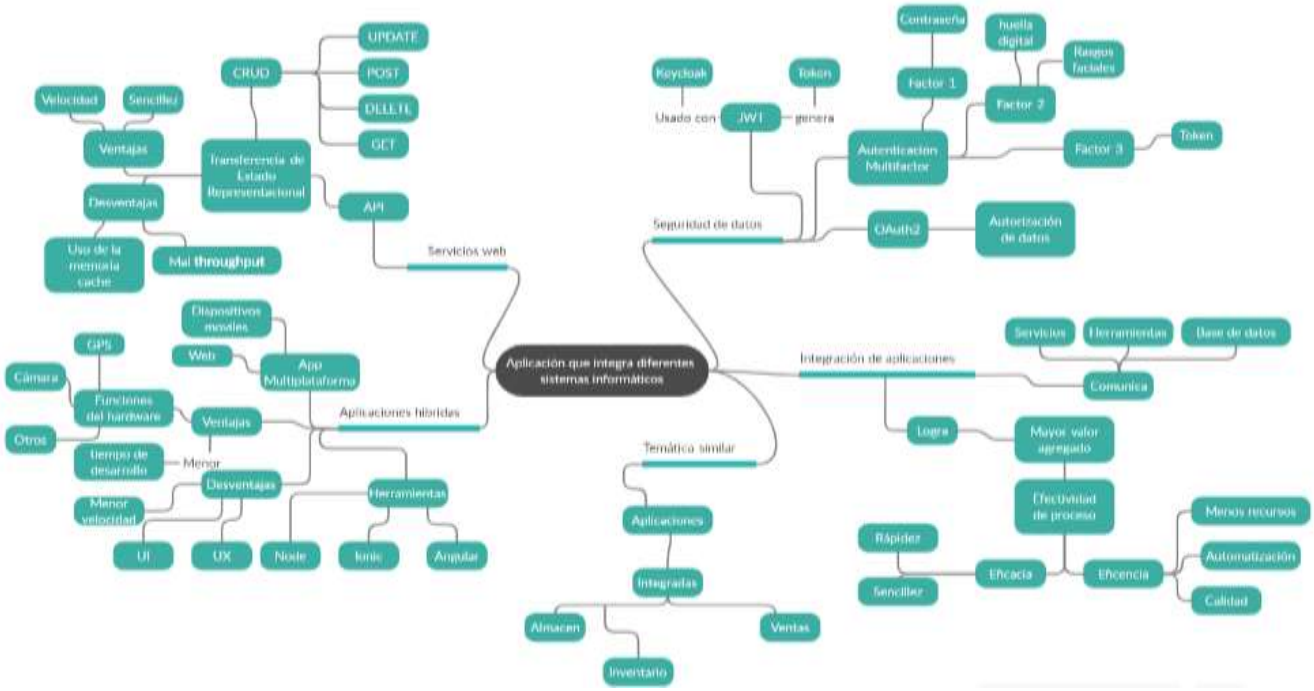


Figura 3: Clasificación de fuente

Fuente: Elaboración propia

En el esquema anterior (mapa mental) se brinda un resumen sobre los puntos que aborda nuestra investigación. Los puntos son los siguientes: Integración de aplicaciones, seguridad de datos, servicios web, aplicaciones híbridas e investigaciones de temática similar.

4.2 PRINCIPALES HALLAZGOS DEL ESTUDIO

En base a toda la investigación realizada, recolectando información de diferentes fuentes bibliográficas de repositorios y bases de datos académicas, presentaremos los principales hallazgos.

4.2.1 Seguridad de Datos

En primer lugar, si se desea hacer un software que administrará datos importantes de la empresa, esta debe ser segura, por lo que a continuación revisaremos lo destacable respecto a la seguridad en la autenticación de los aplicativos propuestos en diferentes investigaciones.

Empezamos, que según Muyón y Montaluisa (2020), sabemos que Rest es la API de servicios web más utilizada y que facilita el intercambio de datos, pero surgen vulnerabilidades que pueden comprometer a la información que ésta gestiona. Por esta razón este artículo nos ayuda con alternativas para mitigar dichas vulnerabilidades con herramientas como el JWT, Keycloak entre otras. JWT es una tecnología que permite transmitir de forma segura información entre las partes con objetos JSON, con Keycloak utiliza JWT para transmitir una llave para dar acceso a los usuarios. Keycloak es un proyecto de código abierto el cual su última versión estable es de abril del 2020, esta brinda la protección de aplicaciones. En este artículo se compara diferentes tipos de comunicación de datos entre cliente y “Web service rest”.

Tabla 2: Herramientas usadas

Herramientas usadas	
Keycloak	Solución para la gestión de acceso, protege las aplicaciones. Ayuda respecto a la autenticación única, multifactor, otros.
JSON Web Token	Estándar basado en JSON, que se utiliza para brindar mayor seguridad a los datos de un software.
Rest API	Sirven para la comunicación de datos. Ejecuta peticiones HTTP que serán aseguradas con JWT
Maven	Utilizado para la gestión de la web
Postman	Esta es un programa usado para probar los servicios web creados
Jboss EAP	Es una plataforma usada para construir y alojar servicios
Eclipse IDE	Es un IDE para el desarrollo de aplicaciones, soporta distintos lenguajes de programación, nos permite la ejecución de pruebas. Se usará la versión 4.8.0 “Photon”

Fuente: elaboración propia

Se observa que hay dos metodologías que menciona Muyón. Por un lado, la autenticación no segura, en el cual se muestra al cliente mandando una petición HTTP al servidor y este último no verifica que la petición sea hecha por un cliente autorizado, por lo que procede a brindar una respuesta. Esto quiere decir que el servidor estaría entregando información confidencial a cualquier usuario sin distinción alguna. Esto varía en gran medida con la segunda metodología mencionada por el autor. Por otro lado, la autenticación segura en el cual se usa la herramienta Keycloak para poder volver más segura la comunicación entre el cliente y el servidor. Esto lo hace realizando configuraciones a su servidor de aplicaciones Jboss y también para toda petición HTTP que llegue se solicitará un JSON web token que valide la autorización para consumir recursos. Esta forma asegurar los datos se prueba primero con software como “Postman” para verificar que todo respecto a los servicios web y la generación de tokens esté funcionando correctamente. Por lo que, al enviar una petición “GET” a un método en específico, no se te mostrará automáticamente los recursos, si no que primero te dará el token y no se podrá visualizar los datos, recién cuando se haya usado el token se procederá a brindar los recursos pedidos con la petición HTTP.

Al realizar la autenticación con token el tiempo de respuesta aumenta considerablemente, pero es igualmente es un tiempo bastante bajo, por lo que sigue siendo una mejor opción sacrificar una pequeña cantidad de tiempo de respuesta a tener la comunicación no asegurada.

Figura 4: Autenticación Simple o Estándar

Fuentes: Muyón

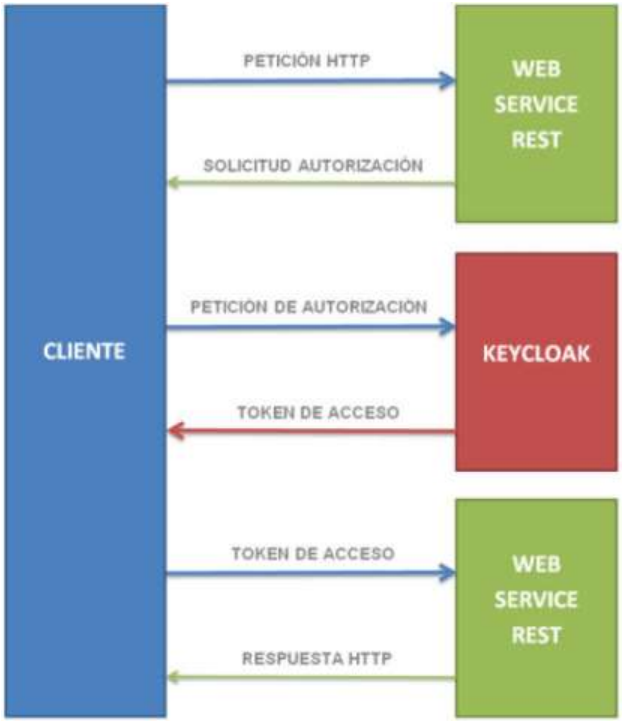


Figura 5: Autenticación con Keycloak

Fuente: Muyón (2020)

Según Ñique (2016), la tecnología, el Internet y la globalización permiten al mundo acceso a diferentes beneficios respecto a productividad, sino también riesgos

referidos a datos privados ya sea de personas o empresas. Por esta razón, como protección ante las vulnerabilidades cibernéticas, es necesaria la aplicación de la seguridad informática.

Se debe adoptar un modelo de defensa para lograr proteger la información y que los recursos estén organizados. Por ello, indica Ñique (2016), que es insuficiente aplicar un inicio de sesión estándar. Es necesario aplicar un inicio de sesión que utilice el doble factor, es decir la autenticación estándar sumado a una contraseña variable. Esta contraseña variable está representada por tokens en físicos o virtuales, SMS o confirmación por correo, preguntas de seguridad, entre otros.

Este método de doble autenticación es un control de seguridad para validar la confidencialidad del acceso a los múltiples recursos. La entidad de gobierno donde se desarrolló el presente trabajo optó por la implementación de la autenticación con doble factor (Ñique, 2016).

En la investigación aplican la autenticación de doble factor para que la seguridad mejore y cumpla con el estándar de seguridad ISO 27001. Asimismo, se detalla diferentes formas de dar mayor seguridad al inicio de sesión. Empresas reconocidas han implementado la autenticación de doble factor ya que han sufrido algún tipo de ataque, esta función la deben activar de forma manual. Por lo antes mencionado creemos que este trabajo nos ayudará a dilucidar algunas dudas referentes a la seguridad de datos.

Se aborda temas como los diferentes factores que pueden usarse para realizar una autenticación. En primer lugar, el primer factor que son valores como contraseñas, nombres de usuario, correo, entre otros. Estos valores son bastante débiles, aunque respecto a la contraseña podría variar la dificultad para descifrar dependiendo de los caracteres que se usen en su estructura. Es decir, si solo se

usan palabras en minúscula el tiempo para descubrirla será de solo unos minutos, al usar mayúsculas y minúsculas el tiempo se amplía a unos pocos días, al añadirle números se descubre en alrededor de un mes y si se le agregan símbolos (asteriscos, entre otros) podría descubrirse en años. En segundo lugar, el segundo factor que son las características únicas que puede tener la persona y poder identificarla, estas pueden ser sus características faciales, huellas digitales, composición de retina, entre otros. Estos métodos de autenticación no son imposibles de replicar con alguna técnica, pero cada vez más se están usando este tipo de factor. Finalmente, hay un tercer factor que se utiliza, este es el uso de tokens físicos y lógicos, actualmente el uso de tokens es bastante popular ya que los ataques informáticos también se han hecho más usuales.

Tabla 3: Tiempo para descubrir una contraseña de terceros

Contraseña	Tiempo para descubrir
divad	Unos pocos minutos
DivaD	Unos pocos días
D9i1v8a2D	Alrededor de un mes
D+9i1v8a2D*	Alrededor de un año

Fuente: Elaboración propia

En esta investigación según los autores Pratama, Linawati y Putra (2018), indican sobre la creación de diferentes sistemas de información web para la mejora de distintos servicios, estos hacen que los usuarios tengan que autenticarse cada vez que van a utilizar un sistema. Por lo que se propone el inicio de sesión único (SSO) basada en token utilizando Json Web Token para los permisos. Esta propuesta

puede unificar los procesos de inicio de sesión existentes en las aplicaciones y poder acceder a estas mediante un solo proceso de “Login”. Además, los usuarios no tendrán el problema de tener que memorizar muchas cuentas. Se comentan comparaciones entre un SSO sin JWT y un SSO con JWT. Siendo que JWT tiene ventaja ya que tiene la capacidad de determinar permisos de recursos cuando un usuario accede al sistema entre otras cosas. En esta investigación se describe el diseño del sistema en el cual nombran 3 fases del protocolo SSO. En primer lugar, establecer el SSO en los sistemas de información, En segundo lugar, es el proceso de registro de los usuarios en el SSO. Finalmente, los usuarios puedan realizar la autenticación mediante un token creado por el SSO.

En esta investigación los autores Viktor, Levendovszky y Ekler (2018) hablan sobre el JWT, sus ventajas y sus desventajas. Asimismo, soluciones que se utilizan para las desventajas. Se comentan varias ventajas en las que destacan dos. Por un lado, mejor escalamiento y rendimiento, esto ya que la confiabilidad de la información de autenticación se transmite a lo largo de la comunicación, en vez de la recuperación del lado del servidor. Por otro lado, la disminución de la complejidad y la descentralización de la autorización gracias al token. Se comentan varias desventajas en las cuales destacan dos. Por un lado, problemas de cierre de sesión, ya que el token no se almacena explícitamente en el servidor, dificultando una posible invalidación ya que el usuario se ha desconectado u otra razón. Por otro lado, el problema de actualización del usuario, los datos del token pueden quedar desactualizados, si es que se realiza una modificación, por esto no se recomienda almacenar datos que cambiarán constantemente en el token. Una solución respecto al cierre de sesión son las fichas de corta duración, es decir que

en el token se incluya la fecha de vencimiento de este y dejar de enviar tokens a los usuarios desconectados.

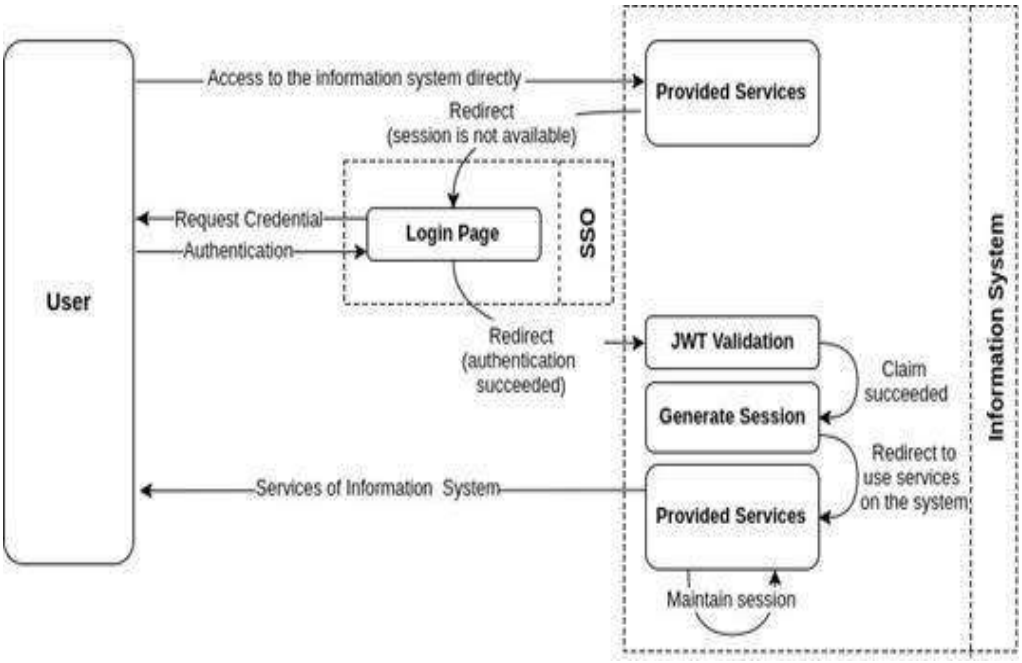


Figura 6: Imagen del sistema JWT

Fuente: Viktor, Levendovszky y Ekler.

Podemos observar que Viktor y Pratama utilizan herramientas similares para poder asegurar sus datos, como lo es JWT el cual se utiliza en las dos investigaciones con el fin de poder tener un SSO o inicio de sesión único. Este se requiere ya que hay muchas aplicaciones dispersas en el que en todas se necesita hacer el respectivo login perdiendo así mucho tiempo en autenticación de datos y también para poder consumir todos los recursos que pedidos.

Según Feria (2018), en su investigación con el fin de obtener la titulación de ingeniería de sistemas “Seguridad para el control de acceso a recursos de la aplicación web de facturación electrónica Openfact, Ahren contratistas generales - Ayacucho, 2017” la empresa de Contratistas Generales posee un software capaz de gestionar comprobantes de pago electrónicos regulados por SUNAT. Luego

surge la idea de distribuir dicha Aplicación comercialmente por lo que aparece la necesidad de tener un control de acceso mejorado para los diferentes servicios que brinda OPENFACT, ya que dichos recursos del software deben también poder ser utilizados desde otros softwares independientes es decir de otras empresas, esta situación demandaba un rediseño del sistema de seguridad. Esta investigación arrojó como resultado el haber logrado implementar capa de seguridad a través del protocolo de seguridad OAuth2, se implementó un eficiente control de acceso en los escenarios en los que el software se usa. Con el fin de mejorar la seguridad Feria Vela realiza configuraciones en el servidor de Keycloack para poder asociar una cuenta de usuario a una cuenta de Google. Además, que también se brinda detalle sobre el despliegue de keycloack para poder implementar OAuth2.

En esta investigación se aborda temas de mejora de seguridad de un software, como el uso de Keycloak y el JSON web token, pero también se implementa el estándar de autorización OAuth2 el cual maneja la autorización de los datos, es decir que gestiona si un usuario está autorizado para poder utilizar un dato en específico. Además, OAuth2 permite que a través de un token se pueda autorizar a un cliente a que utilice el recurso pedido. Esto sin que se tenga que exponer credenciales.

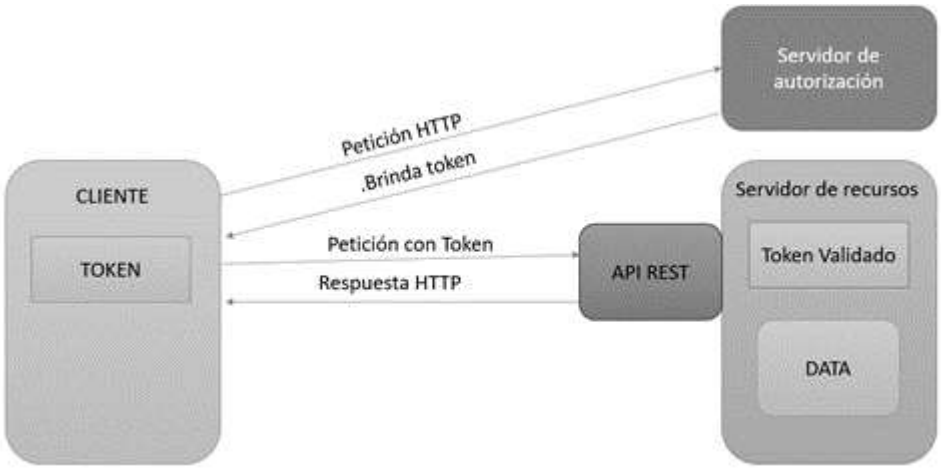


Figura 7: Esquema de autorización

Fuente: Elaboración propia

Según Xu, Jin y Kim (2019), en su artículo “Microservice Security Agent Based On API Gateway in Edge Computing”, buscan lograr una mayor integridad y confidencialidad de los datos que poseen artefactos inteligentes (IoT), por ende, buscan integrar un agente de seguridad para garantizar que solo puedan acceder los usuarios autorizados a la data de los artefactos.

Para ello se integrarán estas plataformas informáticas a una puerta de enlace (API Gateway) para proporcionar un mecanismo de integración. Esta puerta acta ira como un mecanismo intermediario, que realiza la autentificación mediante el uso de JWT, lo que permite la aplicación de control de accesos.

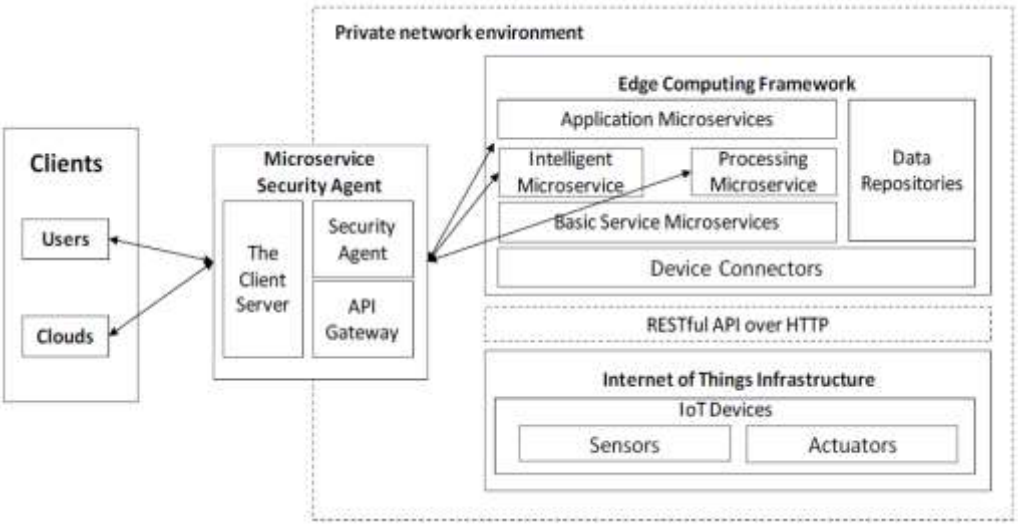


Figura 8: Modelo del sistema

Fuente: Elaboración propia

Se puede apreciar una capa de “clients”, que pueden los navegadores webs o servidores de la nube, la capa de “Edge Computing Framework”, donde se encuentran los múltiples servicios dentro de la red privada a los que se pueden acceder, a diferencia de la nube, estos servicios, se comunican fuera de la red a través de una REST API gracias a la API que cada uno posee.

Este escenario, contiene una interfaz gráfica de usuario, que provee funciones extras como la autentificación de usuario con JWT. Toda petición HTTP get, deberá pasar por dicha Gateway, “Micro service security Agente”, proporciona funciones esenciales para usar el servicio de puerta de enlace API, es el controlador que invoca las funciones necesarias para la respuesta, que permitirá, que la API genere un JWT y una lista de control de acceso (ACL).

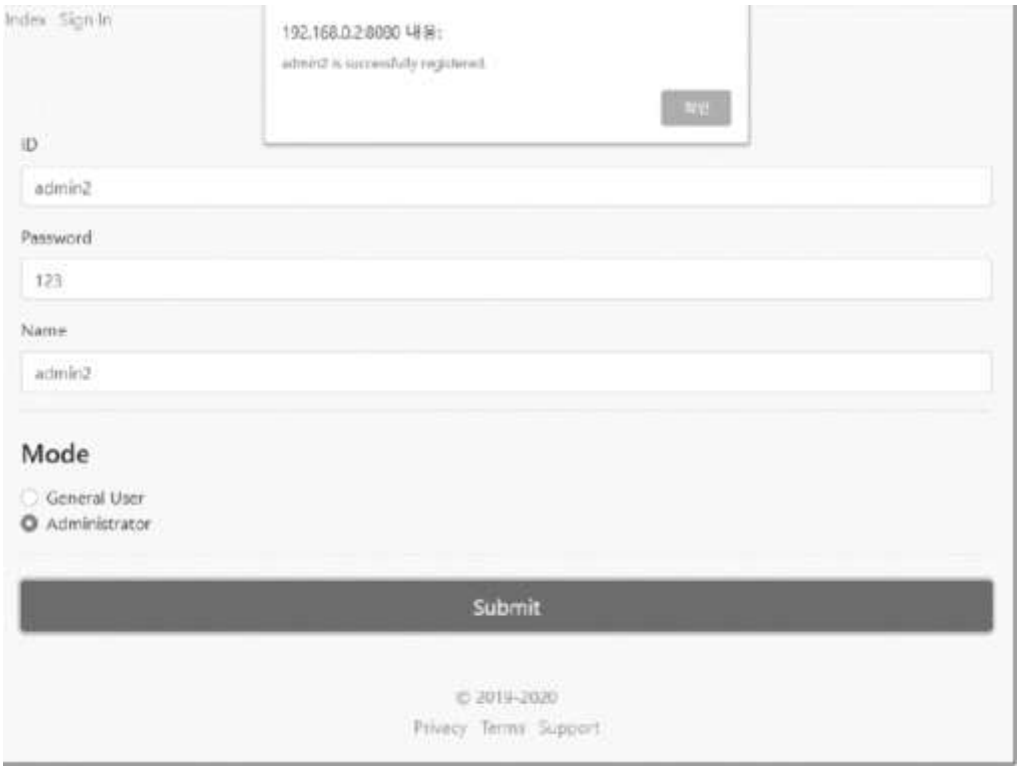


Figura 9: GUI mostrando generación del usuario con éxito

Fuente: Xu, Jin y Kim

Con aquello, podemos remarcar la importancia que tiene la aplicación del JWT, para proveer seguridad adicional a un sistema, como se vio en el proyecto, donde fue vital su uso en el gateway para mantener a los IoT seguros.

Respecto a todas las fuentes revisadas respecto a seguridad de los datos podemos visualizar que se exploran diferentes formas de brindar mayor seguridad. En primer lugar, se comenta sobre el uso del JWT el cual en las investigaciones abordadas se ha implementado con Keycloak una herramienta que nos ayuda con la generación de tokens para una mejor autenticación. En segundo lugar, la técnica del doble factor de autenticación en el cual nos dice que una autenticación común de “Usuario” y “Contraseña” no es suficiente para que los datos privados estén completamente seguros, esta disminución de seguridad se ha dado a través de los años gracias al avance de la tecnología por lo que el doble factor nos ayuda ya que se usará un adicional, este adicional es un token que sólo nos será enviado cuando decidamos iniciar sesión, este token tendrá un tiempo de vida limitado para que no pueda transmitirse a un usuario malicioso. Finalmente, se habla también sobre el uso de OAuth2 el cual se usa para poder autorizar a un cliente si es que puede utilizar un determinado recurso. Esto ocurre con el envío de un token de autorización del servidor de autorización para que luego el cliente lo utilice enviándolo al servidor de recursos, este lo verificará y luego devolverá una respuesta con los recursos pedidos.

Al revisar las diferentes fuentes bibliográficas que abordan el tema de seguridad de datos, podemos darnos cuenta de que es muy importante que en nuestra propuesta de aplicación híbrida tenga una autenticación segura para así proteger los datos confidenciales de la empresa.

4.2.2 API REST

En segundo lugar, una parte esencial para poder manejar datos de diferentes aplicaciones y poder integrarlas es la creación de APIs REST. A continuación, se mostrará lo más importante respecto a cómo diferentes investigaciones abordaron este tema.

Empezando según, Kleiman y Quintero (2018), en su trabajo “GraphQL vs REST: una comparación de rendimiento a nivel práctico” realizan una comparación entre 2 mecanismos arquitectónicos, REST y GraphQL, esta última rediseñando el modelo de búsqueda para que no tome mucho esfuerzo, difiriendo en REST, en su representación, pues sus datos no responden a un URI especificado a diferencia de REST. Por esas diferencias, los autores crean un api (con el uso de Node.js y Express) para cada uno, y comparan las diferencias de eficiencia entre estos.

En las evaluaciones, se realizaron diversas consultas iguales en cuanto la búsqueda de elementos, con cada api y se midió la eficiencia, empezando por el tiempo de respuesta, dando Graphql un promedio de 16 ms, y REST 11 ms, aunque esto se debe a que REST para ser más rápido guarda parte de la URL en el cache, lo cual provoca un consumo de memoria y recursos mayor que Graphql, el cual durante las consultas usó un promedio de 154 bytes, mientras REST uso un promedio de 558 bytes. (Kleiman y Quintero, 2018)

A partir de estos resultados, entendemos que Graphql puede ser una buena opción para aligerar la demanda de recursos, lo cual sirve muy bien en un sistema sobrecargado y enorme, pero para nuestro caso propuesto, nos centramos en 2

sistemas de almacén más pequeños, siendo REST nuestra mejor alternativa, priorizando la velocidad.

Continuando, Según Atencio y Mamani (2017), en su trabajo “Interconectividad basada en APIS REST en aplicaciones web de la municipalidad provincial de Lampa”, estos tienen el objetivo de desarrollar una interfaz que permita que las aplicaciones de la municipalidad se comuniquen. Con respecto a la Municipalidad Provincial de Lampa, esta emplea 6 aplicaciones web para las oficinas de: Recursos Humanos, Planeamiento, Contabilidad, Remuneraciones y Abastecimientos, estas aplicaciones se realizaron en distintos tiempos y con diferentes desarrolladores.

Por ello Atencio y Mamani (2017), indican que es necesario la creación de las API para que los sistemas administrativos se puedan utilizar de nuevo junto con páginas web y aplicaciones móviles nuevas, donde encaja perfectamente la filosofía de REST. Se escoge esta filosofía, debido a los principales proveedores de servicios Web entre los cuales tenemos a Yahoo, Google, Facebook y Twitter, donde se ha dejado de utilizar SOAP y WSDL con el fin de emplear un modelo orientado a recursos para exponer sus servicios y que es más sencilla.

En consecuencia, se plantea en esta investigación un modelo de comunicación de aplicaciones por API REST basados en el estándar JSON, que permite una comunicación fluida y uniformizada sin errores de conversión de datos, soportando incluso formatos multimedia.

Finalmente, por lo antes mencionado creemos que este trabajo nos ayudará a dilucidar algunas dudas referentes: API REST, JSON, Servicios Web.

También, según Santos y Serrano (2017), en su trabajo “Desarrollo de una API REST con sus aplicaciones web y Movil para la venta de ropa Online de la

empresa Roosman” Afirman, que desde hace años el uso de APIs se incrementó. Esto se debe a su aplicación variada, incrementando las experiencias en los “StateHolders”, logrando nuevos modelos de negocio, permitiendo liderar la innovación interna, entre otros. Las API pueden ser aplicadas en diferentes ámbitos financieros, sociales, además en actividades puntuales

Ante esas características, los autores de esta investigación proponen el desarrollo de APIs REST para una tienda online de ropa, ya que se desea incorporar esta solución tecnológica con el fin de lograr competitividad, eficiencia, optimización de costos y un posicionamiento. Por consiguiente, de esta manera obtener una mayor sensación de cercanía con el cliente.

Finalmente, por lo antes mencionado creemos que este trabajo de investigación nos ayudará a dilucidar dudas referentes a: Software para el área ventas, API REST.

A continuación, Henry, Tobias, Vu, Fertig, y Braun (2019), en el artículo de “Pruebas de integración basadas en modelos de sistemas hipermedia”, presentaron una nueva propuesta para el enfoque de testeo y evaluación de una APIS REST, basándose en el modelo, Model-Driven Testing (MDT), que se centra en la ejecución de pruebas a partir de un metamodelo y la validación automática de estas. Aquello, permite facilitar los procesos que usen el paradigma de Ingeniería dirigida por modelos (MDSD), el cual promueve el uso de modelos como artefacto primario, por ello es sensible a cualquier defecto en el modelo, propagando el error a etapas posteriores pues se obtiene código de la transformación del modelo.

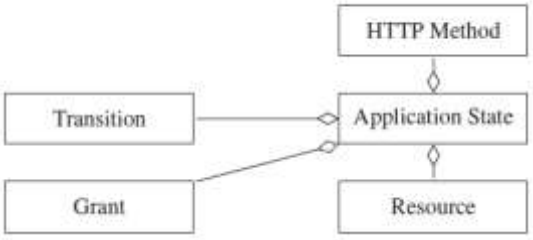


Figura 10: Diagrama UML de un aplication state.

Fuente: Henry, Tobias, Vu, Fertig, y Braun

Por ello MDT genera diversos casos de pruebas a partir del modelo subyacente, dividiéndose en pruebas del lado del servidor y pruebas del lado del cliente. Las pruebas del lado del servidor se dividen en dos fases: análisis estático y análisis dinámico. El objetivo principal del análisis estático es revelar errores en los niveles más altos de abstracción, además este no requiere ejecución de código. Por esta razón se enfoca en el “aplication state” del metamodelo, el cual es un método HTTP y un recurso, que representa una solicitud REST. Con dicha lógica, se puede aplicar herramientas de correcciones automáticas del modelo de la APISREST, que use un autómata infinito no determinado, para la validación, aquel proceso se llamaría ε-NFAcheck. Si faltara una transición a los aplications state del metamodelo, se detectaría un error que sería notificado. Pudiendo validar automáticamente el metamodelo antes de la ejecución de código (Henry, Tobias, Vu, Fertig, & Braun, 2019).

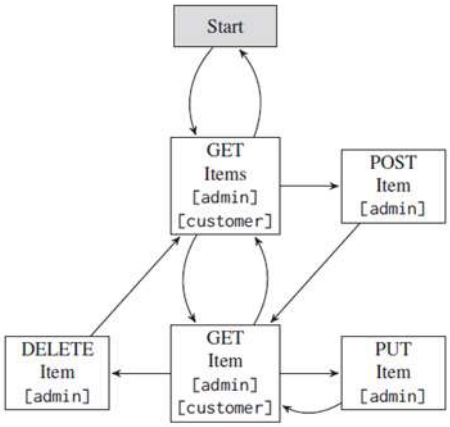


Figura 11: Autómata finito no determinado que representa el funcionamiento de una solicitud REST

Fuente: Henry, Tobias, Vu, Fertig, & Braun

Gracias, a la lógica existente, que plantea para el modelo de un API REST, podemos entender mejor, la lógica que se requiere presentar en el diseño, para que el modelo de solución sea considerado como la una API REST, y cumpla de acuerdo con las funciones necesarias que debería tener, sirviendo esta fuente como punto de guia.

Según, Ažahin y Llkda (2017), presentan el artículo, “Una API web y desarrollo de aplicaciones WEB, para la difusión de información sobre la calidad del aire”, el cual aborda un problema de Turquía, a causa de que se debe medir y analizar constantemente la calidad del aire por el bien de la salud general, por ello se usa equipos especializados que recolectan los datos en un formato preestablecido, Air Quality (AQ), donde se describen los niveles de componentes dañinos encontrados durante las pruebas. Continuando, existen 250 centros de investigación, que recopilan diariamente datos AQ, los cuales son transferidos a una central donde se analizan los datos, y finalmente se almacenan en una web cerrada al público, que

requiere al servidor dentro del Monitoring Center como proxy para la recepción de información.

Figura 12: Arquitectura del sistema de análisis de datos AQ

Fuente: Henry, Tobias, Vu, Fertig, & Braun

Se requiere que la información, sea Open Access, por ello se vuelve a desarrollar el servicio web con la arquitectura REST y permitiendo que los usuarios descarguen los datos luego de consultar la fecha y el lugar, ni necesidad de ningún proxy. El servicio web, se usará JSON, para transferir luego de una consulta, que involucra que el usuario, introduzca la fecha y la ubicación de la estación, para el envío de los datos (Ažahin & Ilkda, 2017).

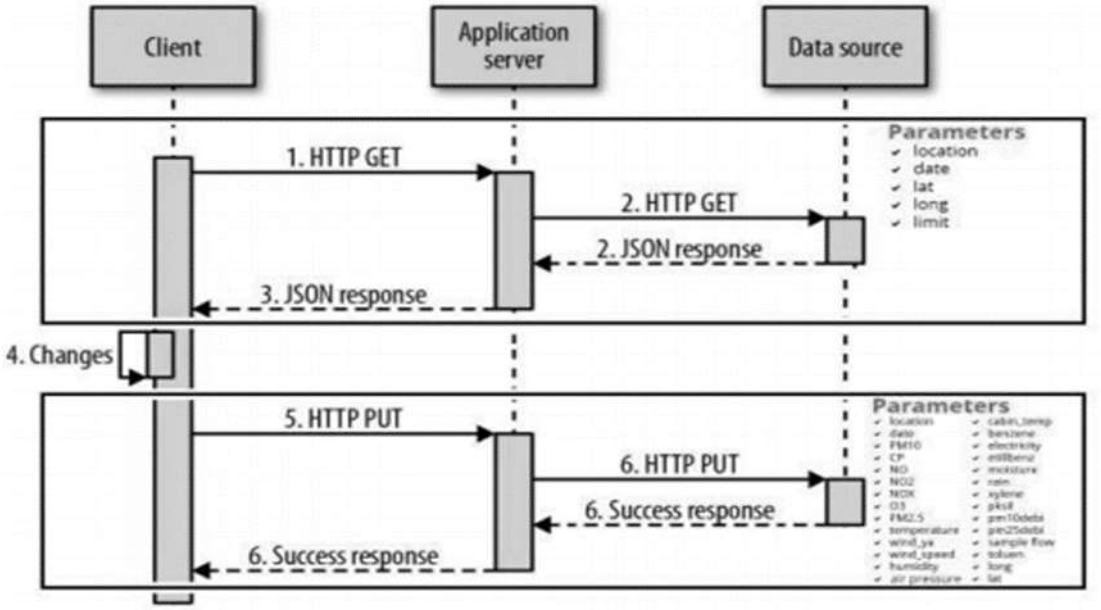


Figura 13: Modelo del envío de datos

Fuente: Ažahin y Ilkda

El trabajo, nos muestra un caso cuyo incidente de falta de compatibilidad con la gente, se soluciona, gracias al uso de una estructura REST, que es más abierta y permite el acceso sencillo, a diferencia de la estructura anterior que requería el uso de proxys, dificultando el acceso. Esto nos ayuda como ejemplo, de la usabilidad de la arquitectura REST, para permitir conexiones sencillas.

Según los autores Jeliazkova, Chomenidis, Doganis, Fadeel, Grasfstrom, Hardy y Kohonen, (2015) en su trabajo de “La base de datos eNanoMapper para información de seguridad de nanomaterial”, tienen el objetivo es desarrollar un servicio, conocido como eNanoMapper, como una herramienta para la BD de NanoSafety. Agregó, que se dispone de la base de datos NanoSafety Cluster, donde se almacena, información ENMS (nanomaterials datos), datos de las partículas de sustancias, donde se pueden extraer identidades químicas y físicas. El formato de estos datos es incomprensible sin el debido procesamiento, debido a

las configuraciones en las que se encuentran guardadas, entre ellas el uso de matrices.

Por ende, se tiene que procesar de diferentes formas según el protocolo en el que se quiera recibir la información, esto es complejo debido a que pueden variar resultados según condiciones experimentales. Por ello el “eNanoMapper prototype database” es parte de una estructura que proporciona nanomateriales y datos experimentales a través de una API de servicios web REST (http: // enanomapper.github.io/API/) y una interfaz de navegador web, usando JSON. La base de datos prototipo eNanoMappe implementa una API REST, lo que permite, realizar llamadas de los datos e importar paquetes de datos, a los módulos que se encargan del modelamiento de estos, además de trasladar también las instrucciones necesarias al debido módulo, para que comience el modelamiento de datos según la metodología y formato que se necesite. Obteniendo el resultado final que se visualizará en una front-end, donde se usa Javascript para presentar los datos relevantes en un formato amigable. Representación de sustancias químicas con composición compleja, y datos experimentales asociados con esas sustancias (Jeliazkova, Chomenidis, Doganis, Fadeel, Grasfstrom, Hardy & Kohonen, 2015).

Figura 14: Modelo de datos

Fuente: Jeliazkova, Chomenidis, Doganis, Fadeel, Grasfstrom, Hardy & Kohonen

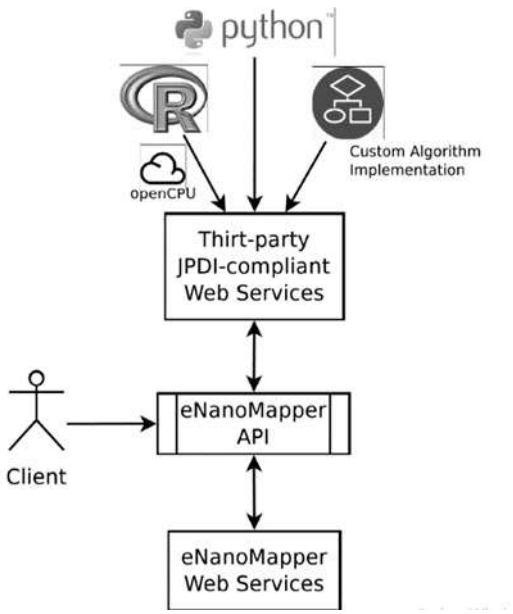


Figura 15: Estructura del servicio

Fuente: Jeliazkova, Chomenidis, Doganis, Fadeel, Grasfstrom, Hardy & Kohonen

Gracias, a estos ejemplos, podemos confirmar la gran capacidad que tiene un APIS REST, para la integración de diversas herramientas y bases de datos, los cuales estaban dispersados en la web. Con su inclusión en uno solo gracias esta APIS REST, se brindó un solo servicio completo, con el objetivo de simplificar un proceso complejo e integrado.

Según, el trabajo de Erdos y Dosztanyi, (2020), llamado “Analyzing Protein Disorder with IUPred2A", se desarrolla un servicio web para resolver una problemática, pues existen genomas con proteínas desordenadas y regiones proteicas (IDP / IDR), para el estudio de la parte desordenada se necesitan herramientas computacionales para comenzar la caracterización de regiones desordenadas.

IUPred es uno de los métodos de predicción más utilizados, junto con ANCHOR, que se enfocan en estudiar y estimar las interacciones favorables entre aminoácidos de dichas zonas. Con la combinación de ambas herramientas se crea IUPred2A, Por ello, para su uso, se debe entrar a la página web, y proporcionar un identificador UniProt o una secuencia de proteínas en el cuadro de entrada, seleccionar algunos filtros, especificar el formato, y luego un panel gráfico de la página mostrará un gráfico de predicción correspondiente a la secuencia proteica enviado (Erdos & Dosztanyi, 2020).

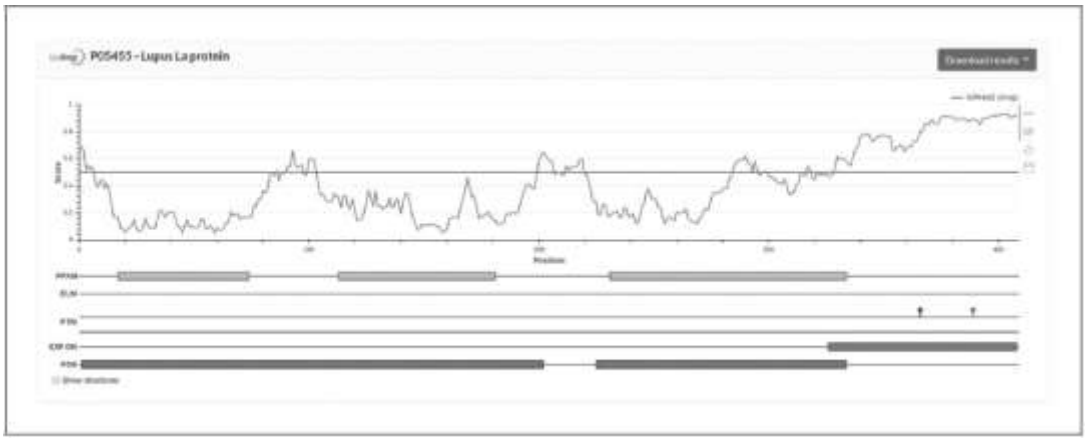


Figura 16: Resultados gráficos del servicio web

Fuente: Erdos & Dosztanyi

También, la herramienta permite su uso por medio de APIS REST, enviando datos en formato JSON. Con el uso de Phyton, se puede enviar URL a las herramientas, para su uso ejemplo: La variable “response” recibe los datos, de procesar con el formato ANCHOR, la proteína “Q32P44”

Requirements

Python3.6, python3-requests library version 2.22.0, python3-json library version 2.0.9

Sample code

```
import requests
import json
URL = "https://iupred2a.elte.hu/iupred2a/anchor/\
  q32p44.json"
response = json.loads(requests.get(URL).text)
print("\n".join(["{}\t{}".format(response["sequence"]
  [n], x) for n, x in enumerate(response["anchor2"])]))
```

Figura 17: Código de ejemplo para el uso de la API del servicio

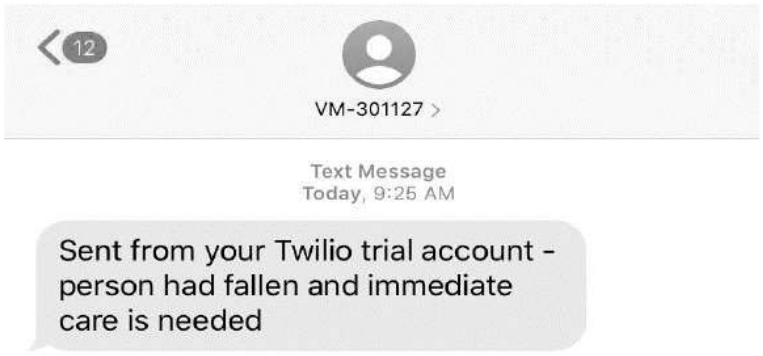
Fuente: Erdos & Dosztanyi

De esta fuente, podemos relacionar el uso de una APIS REST, con JSON, para la conexión y envío de información a cualquier programa que desee trabajar con el Sistema. Osea en este caso, la API permite que se pueda trabajar en esta siendo llamada por cualquier sistema con su consulta URL respectiva, de ese modo podemos observar que la API puede ser llamada incluso por un programa, desde el mismo código.

Según, Sreenidhi y Satyanarayana, (2020) en su proyecto “Real-Time Human Fall Detection and Emotion Recognition using Embedded Device and Deep Learning”, Se desarrolla una inteligencia artificial, que usa CNN (Red neuronal convolucional) para detectar en un video cuando una persona, sufre una caída y notificar de forma

inmediata a través de un mensaje SMS con una APIS REST, proveniente de un servicio llamado Twilio.

El proceso consta de equipar la unidad para la detección, procesar la información, detectar la caída y finalmente enviar la alerta de emergencia. Para ello la camara se



conecta mediante cable LAN a una computadora, para las pruebas (aunque podría hacerse vía bluetooth). Si se detecta una caída, la señal se envía con el uso del servicio Twilio, que provee una API REST con JWT, conectados a los servidores necesarios para mandar un mensaje SMS (Sreenidhi & Satyanarayana, 2020).

Figura 18: Envió de SMS, al celular, luego de detectar una caída.

Fuente: Sreecidhi y Satvaaravana

```php
<?php

// Update the path below to your autoload.php,
// see https://getcomposer.org/doc/01-basic-usage.md
require_once '/path/to/vendor/autoload.php';

use Twilio\Rest\Client;

// Find your Account Sid and Auth Token at twilio.com/cons
// DANGER! This is insecure. See http://twil.io/secure
$sid    = "ACXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX";
$token  = "your_auth_token";
$twilio = new Client($sid, $token);

$message = $twilio->messages
                  ->create("+15558675310", // to
                           [
                               "body" => "This is the ship
                               "from" => "+15017122661"
                           ]
                  );

print($message->sid);
```

Figura 19: Código PHP, para la conexión con el API SMS de Twilio

Fuente: Sreecidhi y Satvaaravana

Aquí podemos observar, como incluso existen servicios que brindan acceso a una APIS REST ya preparada, la cual requiere una leve personalización, para su uso en algún sistema, según la empresa, en este caso de Twilio, para el sistema de mensajes SMS, con ello podemos observar que las soluciones de API REST, son una opción rentable y solicitada, que incluso existen negocios enfocadas en el préstamo de estas.

Según, Cornetta, Mateos, Touhafi y Muntean (2019), en el proyecto “Design, simulation and testing of a cloud platform for sharing digital fabrication resources for education”, promover la educación científica, se usa la metodología de Fab Lab, de exponer al estudiante a múltiples instrumentos tecnológicos. Pero la implementación de un Fab Lab es costosa, por ende, se desarrollará el sistema “NEWTON Fab Labs” como arquitectura para el acceso remoto a un Fab Lab como servicio en la nube, el cual consta de tres niveles, osea es un Hub-and-Spoke. Además, se usa un Cloud Hub, que mantiene un registro de las conexiones y sirve como enrutador para la transmisión de solicitudes, lo cual centraliza el sistema (Cornetta, Mateos, Touhafi & Muntean, 2019)

También, cada Fab Lab, cuenta con un “Fab Lab Gateway”, funcionando como una especie de envoltura para el laboratorio, el cual gestiona información de las máquinas, y crea un servicio para cada una de estas para poder ser usadas a través de una API REST, que confirma la identidad del usuario con JWT, pues sería riesgoso conectar la máquina directamente a internet (Cornetta, Mateos, Touhafi & Muntean, 2019).

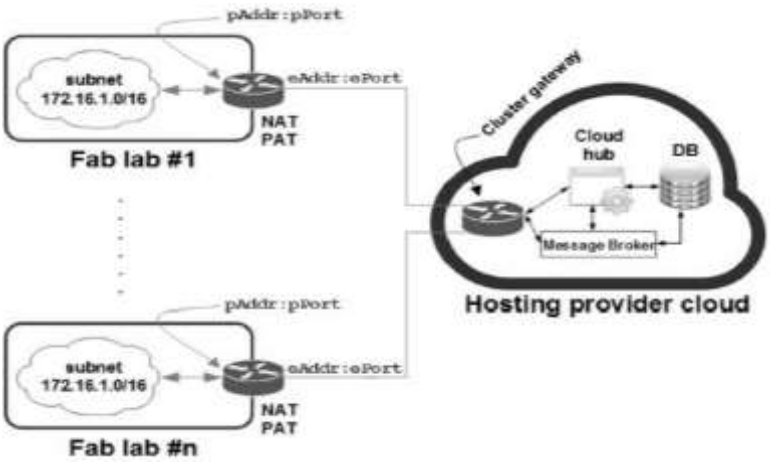


Figura 20: Arquitectura del sistema simplificada

Fuente: Cornetta, Mateos, Touhafi y Muntean

Vemos como la conexión API REST, tiene uso dentro de un sistema más complejo, siendo de este modo parte del sistema, como un complemento que permite integrar el token, a un sistema ya funciona, de este modo podemos, relacionarlo con nuestra propuesta, en lo que respecta que, a los sistemas ya funcionales, le aumentamos una APIS REST, para su mejora.

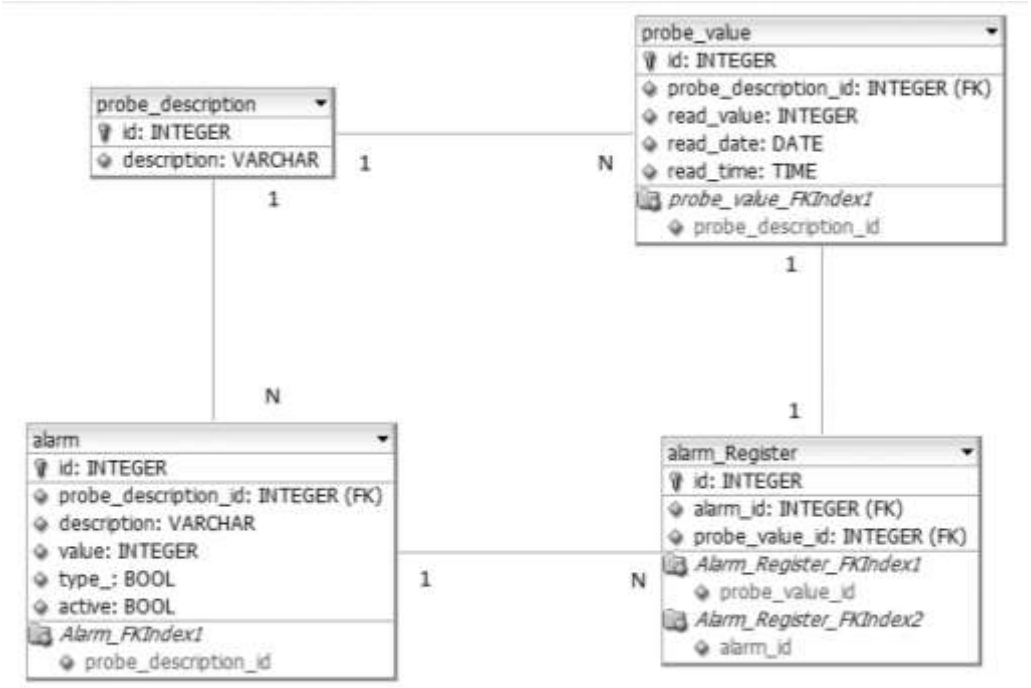


Figura 21: Diagrama de entidades

Fuente: Cornetta, Mateos, Touhafi & Muntean

Se usan las siguientes entidades para guardar la información

- Probe-description: almacena la descripción del sensor y su identificador
- Probe-value: Almacena las lecturas del sensor, con su respectiva fecha.
- Alarm: Almacena la configuración de la alarma, para que se active o no.
- Alarm_register: Almacena los registros de alarma cuando ocurre uno

La aplicación, almacenará los datos en la base de datos PostGreSQL, al pasar por la aplicación web que contiene la API REST.

Se ha visto, como la inclusión de una API REST, permitió automatizar un proceso entero, mejorando el rendimiento de este, lo cual buscamos lograr también con nuestra propuesta, de automatizar algunos procesos para la empresa, con la integración de sus sistemas.

Según Vamshi y Ganapathy (2019), en el proyecto “Mobile Application for Uploading Marks and Student Attendance Management System“, Se desea elaborar una aplicación móvil, para el teléfono del docente en una universidad. El instructor toma asistencia en el teléfono y la aplicación realiza el procesamiento de los datos marcados por la facultad y también los datos se guardarán en el servidor web.

Por ello, para el desarrollo de la aplicación en el back-end usa las tecnologías de: Express.js y Node.js, que permitirán lograr una conexión API con métodos URL (Api REST) de la aplicación con una base de datos de MySQL que contendrá los datos del estudiante y un registro de las clases a las que asistieron. Siendo la API de Express.js, la interfaz. El login, permitirá que un docente pueda iniciar sesión para iniciar el registro donde marcará, que alumnos se encuentran y cuáles no. Estos por

su parte al iniciar sesión como alumnos, podrán ver su registro de asistencias. (Vamshi & Ganapathy, 2019).

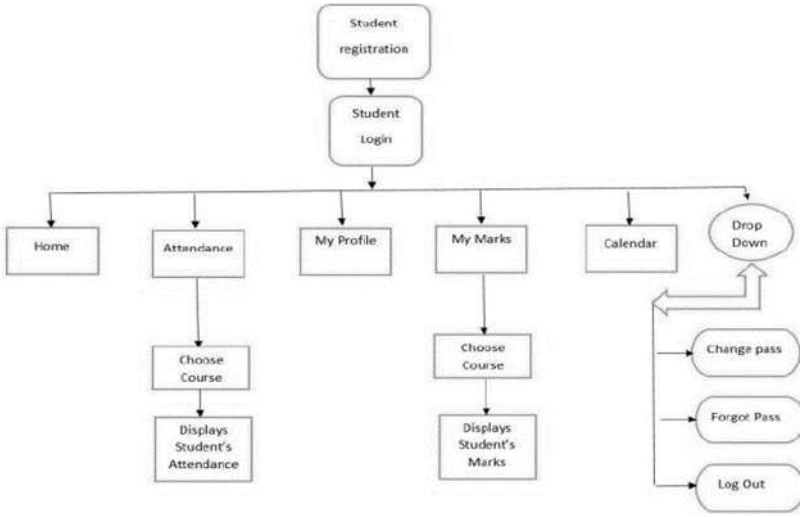


Figura 22: Pantalla principal y la vista frontal una vez que el Estudiante inicia sesión

Fuente: Vamshi & Ganapathy

En este trabajo, para la creación de la API, se usa Node.JS y el framework de Express.js, los cuales son las tecnologías que nosotros proponemos para la creación de la APIs REST.

Según Alonso (2017), en su investigación para obtener el grado de ingeniero informático “API REST y sistema de aprovisionamiento en containers para servIoTicy” En la presente investigación aborda a detalle el desarrollo en Node.Js de una API REST y con el uso de Docker containers y Rancher se lleva a cabo la integración de un sistema de aprovisionamiento. Además, se procede a detallar temas como la planificación y el presupuesto de las tareas realizadas.

Con el fin de actualizar la plataforma servIoTicy se procede a usar nuevas tecnologías que permitan lograr un aumento del rendimiento. Para ello se realizan

un cambio de la API REST, se procede a integrar el sistema de aprovisionamiento en Containers, luego se evalúa la nueva API basada en Linux Containers.

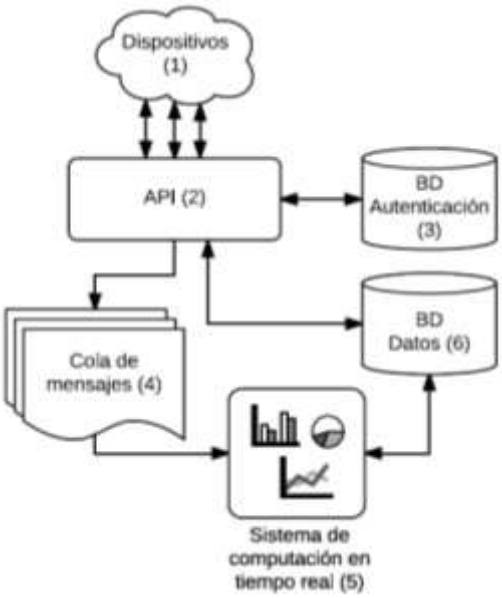


Figura 23: Estructura de la Base de dato

Fuente: Alonso

En la renovación del API REST se destaca el uso de Node.js, cuyo paradigma de programación es distinto a los modelos tradicionales, y también la tecnología que se usara para que la APIS propuesta pueda ejecutarse y verse beneficiado directamente en el rendimiento de la aplicación.

Según Wang, Lai, Xia, Wang y Yang (2019), en su proyecto “A Parallel Computation and Web Visualization Framework for Rapid Large-scale Flood Risk Mapping”, se busca desarrollar un servicio web, que ofrezca una simulación de la forma más rápida posible en cierta zona propensa, para de este modo poder desarrollar planes, pero existe la dificultad de que para ello se tiene que pasar una gran cantidad de datos provenientes de ciertos equipos para el monitoreo del área, lo cual dificultad la transformación de datos, a continuación se mostrará la estructura para el mapeo del servicio web que se propone.

Para el Mapeo de la red, contiene 4 niveles, “flood prediction and calculation”; “data transformation to web GIS”; “A web service layer”; “And Web visualization on the client”, que en un conjunto permiten que de los equipos que capturan los datos del terreno, se llegue con los datos transformados en información, a un usuario por medio del servicio web (Wang, Lai, Xia, Wang & Yang, 2019)

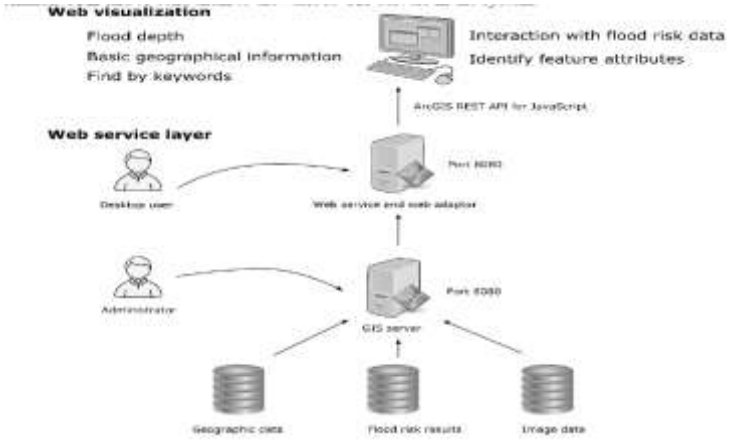


Figura 24: Arquitectura de la visualización web y capa del servicio web

Fuente: Wang, Lai, Xia, Wang y Yang

A continuación, me enfocare en los 2 últimos niveles. Para acceder y procesar los datos en la web, se usa una interfaz REST API para interactuar con los datos servicios y para proporcionar aplicaciones de mapeo web ricas en datos.

Para facilitar este mapeo, el servicio web usara para la conexión una API ArcGIS para JavaScript. Esta es una librería para poder decodificar datos y presentarlos de una forma gráfica, en este caso asimila los datos geográficos y crea 2 capas en paralelo, del terreno, y de la inundación (Wang, Lai, Xia, Wang & Yang, 2019)

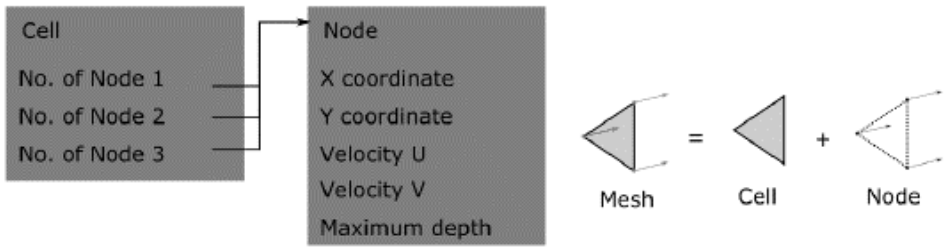


Figura 25: Conversión de datos a geométrico

Fuente: Wang, Lai, Xia, Wang y Yang

Se puede apreciar, que una API además de transferir la información, puede incluir librerías que permitan la transformación de la información que envía y recibe, permitiendo la realización de algún proceso más específico, expandiendo las posibilidades y utilidades que presenta una API REST.

Según, Darzo, Yamate, Yamada y Kelson (2019), en su proyecto “FuncTree2: An interactive radial tree for functional hierarchis and omics data visualization”, en la biología, concretamente en los estudios de omics, las herramientas actuales que transforman los datos en información jerárquica, se volvieron muy limitado a la hora de plasmar información más compleja, como tendencias, perdiendo información.

Por ello, desarrollan el servicio de FuncTree2, una aplicación web fácil de usar que convierte las clasificaciones jerárquicas en árboles radiales interactivos y altamente personalizables, debido a que usa para la transferencia de información el formato JSON, a través de una REST API, cuyo código fuente está disponible en GitHub y permite a los investigadores visualizar sus datos simultáneamente en todos sus niveles, evitando las limitaciones de las anteriores herramientas (Darzo, Yamate, Yamada & Kelson, 2019).

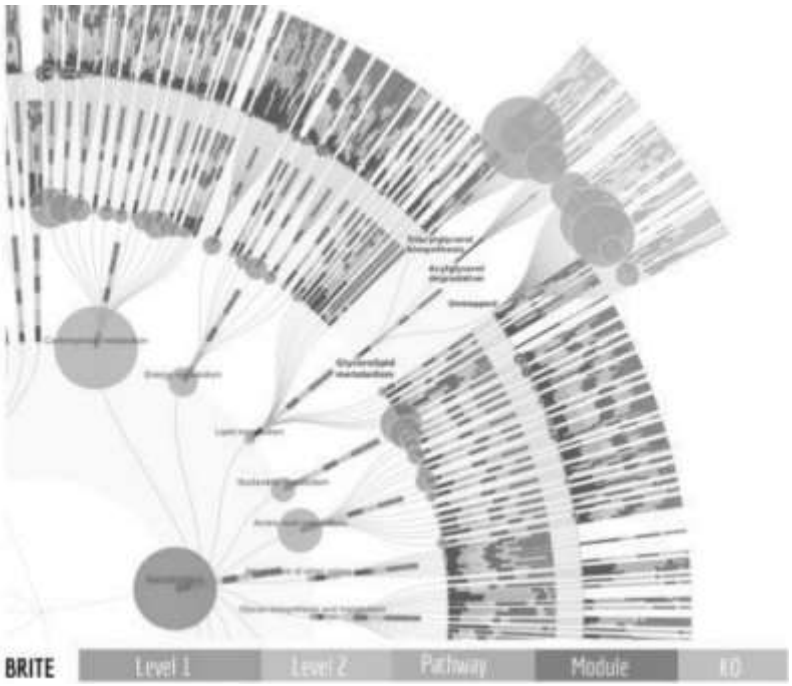


Figura 26: Gráfica de la información del omic.

Fuente: Darzo, Yamate, Yamada y Kelson

Volvemos, a observar por segunda vez una API, de código abierto, que permite la integración de sus funciones a otros programas, remarcando la facilidad que estos tienen para la integración de programas.

Según, Kim, Choi y Sun, (2018), en su trabajo “BRCA-Pathway: Un sistema de integración y visualización estructural de datos de cáncer de mama TCGA en las rutas KEGG”, el cual propone el diseño de un sistema BRCA, para detectar el cáncer de mama, para ello se requiere diversos análisis de diferentes datos, cada análisis requiere su herramienta, y los resultados se deben comparar con otros para la detección del cáncer, por ejemplo, variación del nucleótido (SNV) y la variación del número de copias (CNV), para encontrar ambas variaciones se requiere diversos datos, y realizar un proceso para cada una.

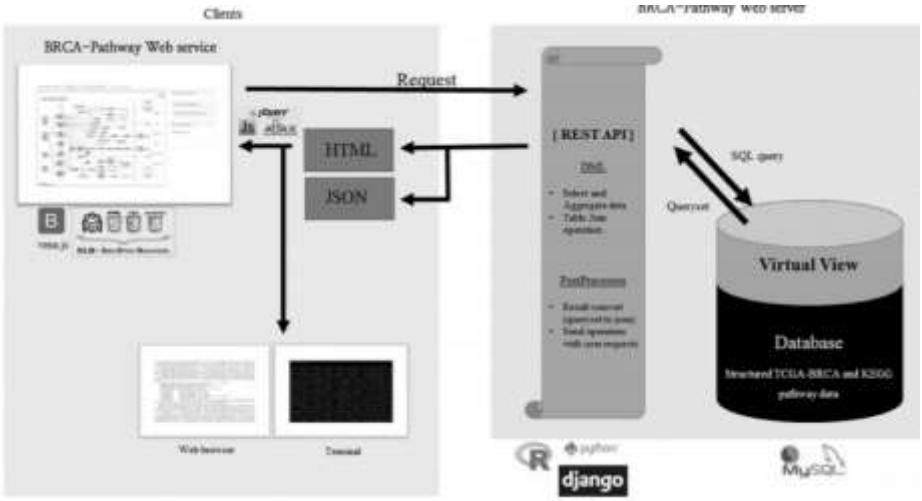


Figura 27: Estructura del sistema BRCA

Fuente: Kim, Choi y Sun,

Por lo tanto, se requiere un servicio que permita la detección de este tipo de cáncer, debido a esto se necesita hacer el proceso completamente en vez de segmentarlos, mediante una integración de estas herramientas. Por ello, se optó por realizar un sistema BRCA-Pathway, el cual se compone de 3 componentes: base de datos, REST API, y el front-end de la web. Aquel sistema, sigue el protocolo TCGA para el formato de los datos de las células cancerígenas, además de contar con diversas herramientas integradas, para el análisis de cada proceso necesario en el mismo sistema, usando APIS REST para extraer los datos conseguidos, además de guardar los datos de las operaciones ejecutadas (Kim, Choi & Sun, 2018).

De este modo, podemos volver a ver, como se usa la APIS REST, para integrar diferentes herramientas informáticas, para simplificar un proceso, gracias al uso conjunto de estos.

4.2.3 Aplicación Híbrida

Una parte esencial para la propuesta a realizar es que la solución se pueda usar en diferentes plataformas, por esta razón se ha tomado en cuenta las aplicaciones híbridas. A continuación, se mostrará lo más importante respecto a cómo diferentes investigaciones abordaron este tema.

Según Molina (2017), en su trabajo “Desarrollo de módulos de funcionalidad para aplicación móvil utilizando tecnologías de desarrollo híbrido para la empresa BSD Enterprise S Enterprise S.A de C.V” BSD Enterprise está elaborando una aplicación móvil, la cual se centrará en el desarrollo de módulos con tecnología híbrida. EL desarrollo de aplicaciones híbridas en cuanto velocidad es mejor generalmente que los desarrollos en tecnologías nativas, pero además cuenta con las ventajas de ser multiplataforma.

Por ello, a corto plazo los módulos híbridos ayudan en la disminución de tiempo y con ese extra añadir nuevas funcionalidades, claro esto es mejor si ya se tiene algún módulo híbrido implementado anteriormente. Aunque, en el futuro la tecnología híbrida se cambiará por una nativa con el fin de lograr una mejor UX y rendimiento (Molina, 2017).

También, se dio uso de “Ionic” para la creación de las interfaces utilizando HTML. Se utilizará webView ya que los módulos no necesitan alto rendimiento. Otro framework que utilizaron fue Angular.js. Asimismo, en esta investigación se comenta bastante sobre la organización para realizar el proyecto y sobre la metodología scrum y sus diferentes artefactos (Molina, 2017).

Tabla 4: Alternativas de solución

Alternativas de solución

React Native	Ionic	NativeScript
Framework para crear interfaces de apps móviles	Framework para desarrollo de aplicaciones híbridas, hecha sobre Angular y Cordova	Framework de código abierto para desarrollo de aplicaciones
No usa webViews ya que no usa HTML. Hace posibles interacciones sin utilizar herramientas como Apache.	Ionic proporciona un conjunto de directivas desde Angular (como elementos HTML personalizados)	Similar a React Native, no usa webViews para las interfaces. Usa XML y Javascript.

Fuente: Elaboración propia

Podemos notar que al decidir realizar módulos nuevos para implementarlos en su aplicación nativa y llevarlo a cabo tuvieron distintos inconvenientes. Por un lado, el hecho de usar un framework como Ionic que, aunque tenían experiencia utilizándolo y se les hacía sencillo desarrollar con dicho framework, surgieron inconvenientes como errores en el diseño, bugs inexplicables, problemas “issues” que no se habían arreglado hasta esa fecha y problemas de rendimiento. Algunos de estos problemas al utilizar Ionic deben haber sido resueltos a lo largo de estos 3 últimos años, por lo que, si optamos por utilizar Ionic como herramienta de desarrollo de nuestra propuesta de aplicación híbrida, seguro no encontraremos muchos de los errores mencionados anteriormente. Por otro lado, se comenta que las mayores complicaciones fueron realizar las integraciones de los diferentes módulos híbridos a la app nativa ya existente, ya que en varias no pudieron probar

el buen funcionamiento de sus módulos porque aún no se había hecho la integración. Además, que no encontraron demasiada documentación sobre cómo realizar dicha integración.

Según Angulo (2013) cada vez está aumentando el uso de celulares smartphone, esto es algo que actualmente en el 2020 se afirma ya que casi todas las personas pueden adquirir un smartphone, siendo que hay diferentes gamas de dispositivos cada uno con un rango de precio distinto para que sea accesible a la mayoría de las personas. A mayor gama de celulares mejores funcionalidades puede tener. Las aplicaciones creadas para estos dispositivos también son una cantidad inmensa, aunque algo que se está buscando actualmente es poder hacer aplicaciones que puedan ser usados en cualquier dispositivo sin importar su SO. Esto conduce a que cada vez más se quieran hacer aplicaciones híbridas ya que estas logran ser multiplataforma y disminuye notablemente el tiempo de desarrollo respecto a una aplicación nativa o una web. Además, pueden ser subidas en la tienda de aplicaciones como si fuese una aplicación nativa. Asimismo, trata de usar lo mejor de las apps nativas como el uso del hardware del dispositivo (GPS, cámara, entre otros) a comparación de las webs que tienen el uso del hardware extremadamente limitado.

Como Angulo bien dice las aplicaciones híbridas pueden usar los recursos hardware del dispositivo no en su totalidad, pero sí lo suficiente para poder realizar diferentes funcionalidades útiles para los usuarios. Algo que diferencia a las páginas web de las aplicaciones híbridas es que estas últimas no dependen de estar conectados a la red de internet.

Las aplicaciones híbridas son una gran opción para realizar por diferentes razones. En primer lugar, si en la propuesta del proyecto no contiene requerimientos que necesiten usar muchos recursos hardware del dispositivo

entonces se puede hacer el proyecto de forma híbrida para que pueda usarse en diferentes dispositivos y cuente con muchas otras características de las aplicaciones nativas. En segundo lugar, el costo de desarrollo es mucho menor ya que no se necesitará realizar la misma aplicación para otro SO como IOS, solo crea una aplicación y se podrá abordar todos los dispositivos. Además, el rendimiento también es cercana a una nativa. Finalmente, si se necesita que el proyecto tenga presencia en Google Play, AppStore, entre otros y sin crear una app nativa entonces una aplicación híbrida.

No solo son beneficios lo que tiene una aplicación híbrida, también tiene algunas desventajas notables. Por un lado, el diseño de Interfaz de usuario y de experiencia de usuario se ve bastante afectada ya que estas no tendrán el mismo patrón que los usados en SO en específico, es algo que sí se puede obtener al hacer una aplicación nativa. Por otro lado, a comparación de las nativas las híbridas son lentas respecto a la ejecución, si se desea que ejecute gráficos 3d o muchos cálculos en tiempo real, este tipo de app puede cumplir su función, pero no es lo más recomendable.

Al ver todos estos puntos de comparación entre los 3 tipos de aplicaciones, podemos escoger como mejor opción la creación de una aplicación híbrida gracias a que nos ahorrará una gran cantidad de tiempo y no se utilizará muchos recursos hardware del dispositivo. Además, que no vemos como inconveniente que el diseño de nuestra interfaz sea un poco distinto a los estándares dados en SO específicos, igualmente se dará una interfaz que de una buena experiencia de usuario.

Para Fraga (2017) indica que Imatia Innovation es una empresa de desarrollo de software. Desarrolla un sistema que sigue la arquitectura cliente-servidor que le permite llevar un control de las horas de trabajo de los empleados.

El servidor ofrece un conjunto de métodos pensados para ser utilizados por otro software (API o Application Programming Interface) y utiliza REST.

El único cliente implementado se trata de una aplicación móvil híbrida. Este tipo de aplicaciones utilizan frameworks de desarrollo basados en lenguajes web como HTML, CSS o JavaScript, que posteriormente se pueden empaquetar para varios sistemas operativos móviles como Android, iOS o Windows Phone. Tienen la ventaja de que permiten reducir el tiempo de desarrollo ya que, en el caso de las aplicaciones nativas, habría que realizar una para cada plataforma. Este trabajo tiene el objetivo de ampliar funcionalidades al sistema y, además, prepararlo para su futura evolución (Fraga, 2017).

Finalmente, por un lado, se ha añadido un nuevo apartado que muestra una serie de gráficas y estadísticas para que el usuario pueda llevar un control de sus eventos y detectar posibles errores. Se muestran datos del número de entradas y de salidas, el número total de horas trabajadas durante la semana vigente y una gráfica que representa las horas trabajadas durante cada día laborable de la misma semana. Para la creación de las gráficas se ha utilizado una librería llamada ng2-charts. Por otro lado, se ha creado un nuevo servicio que permite registrar eventos basándose en la tecnología IBeacons. Con todo, el resultado es un sistema versátil y escalable que podrá ser ampliado en el futuro (Fraga, 2017).

Según Vilalta (2019), en su investigación “Desarrollo de Partyfy, una app híbrida en React Native” se realiza una aplicación híbrida para una discoteca, en esta aplicación se desea que pueda mostrar la ubicación de los diferentes locales de la discoteca como también poder brindar la facilidad de que puedan saber en qué locales habrá algún evento y la venta de las entradas de los dichos eventos. La app está hecha para que desde cualquier dispositivo el usuario que desee

algún tipo de entretenimiento puedo ubicarlo en dicha app y poder ir. Este proyecto se realizará con React Native para la realización de una app híbrida, ya que les permite aprovechar los diferentes puntos positivos de las apps nativas como de las páginas web.

Este proyecto nos ayuda en el desarrollo de nuestra investigación ya que nos brinde una variedad de pruebas que se realizan para examinar la usabilidad y otros puntos de la aplicación. Además, nos brinda detalle sobre el proceso de desarrollo del prototipo, desde los wireframe hasta la interfaz de usuario final. Asimismo, se brinda a detalle todos los requerimientos.

Figura 28: Vista de la aplicación

Fuente: Vilalta

Como podemos observar este el proyecto de Vilalta trata del desarrollo de una aplicación híbrida, por lo que se realiza una comparación entre los diferentes tipos de aplicación, se escoge realizar una aplicación híbrida ya que esta también permite la manipulación del recurso de GPS por lo que no es necesario realizar una aplicación nativa.

Algo interesante que se usa en esta investigación es un pequeño esquema donde se clasifican los requerimientos en 4 secciones “must”, “should”, “can”, “won’t” llamado esquema “MoSCoW”, es bastante intuitiva para usarse y ordenar que requerimientos se deben hacer obligatoriamente, cuales se tomarán en cuenta y cuales no se harán.

También se enfoca en utilizar elementos del logo de la empresa para extraer la paleta de colores y la tipografía he implementarlo en sus prototipos algo que de verdad ayuda a que la aplicación tenga mayor personalidad. Además, para mejorar la experiencia de usuario se realiza algo que a mi parecer debe hacerse más seguido para poder obtener mejores resultados en la construcción de la UI, esto es un planteamiento de que tipos de persona descargarían he usarían la aplicación y respecto a esto poder adecuar las interfaces de usuario y sea más fácil para ese público poder utilizarlo.

Revisando esta investigación nos damos cuenta de que debemos poner atención en cómo dar un buen orden a nuestras carpetas de archivos de código, con el fin de que todo el desarrollo se pueda llevar de una forma más ágil, en el caso de Vilalta realiza un orden de carpetas el cual es: componentes, navegación, idioma, pantallas, Utils, Assets (recursos).

Este proyecto nos ayuda en el desarrollo de nuestra investigación ya que nos brinda una variedad de pruebas que se realizan para examinar la usabilidad y otros puntos de la aplicación. Además, nos brinda detalle sobre el proceso de desarrollo del prototipo, desde los wireframe hasta la interfaz de usuario final. Asimismo, se brinda a detalle todos los requerimientos.

Según Arias (2017), en su investigación “Comparación de la Usabilidad de una Aplicación Web una Aplicación Híbrida en Dispositivos Móviles” ha hecho una

comparación entre la usabilidad de una aplicación web y una aplicación híbrida en dispositivos móviles. Para este fin Arias realiza un prototipo software para el envío de reclamos estudiantiles de una universidad, esto mediante técnicas de diseño centrados en el usuario. Este software hecho por Arias se comparará con una aplicación web realizada por la misma universidad el cual tiene como fin cumplir el mismo objetivo respecto a los reclamos de estudiantes. En la investigación se detalla desde la creación de prototipos como wireframe hasta la versión final. Arias, considera que es conveniente el desarrollo de una aplicación híbrida con el fin de que se mejore la experiencia de usuario, esto basándose en sus resultados. En cuanto a la percepción de utilidad del sistema resulta que no es muy significativa la diferencia entre los dos tipos de aplicaciones. Respecto a la percepción de calidad de información se encuentra una diferencia significativa a favor de la aplicación híbrida siendo que se usó la misma información en las dos aplicaciones. Uno de los puntos fuertes en la aplicación hecha en esta investigación es la calidad de la interfaz gráfica. Además, de que la tasa de éxito y tiempo mejoraron bastante al usar la aplicación móvil, respecto a la web. Esto se debe a los diferentes patrones de diseño usados para que hubiese mayor comprensión de lo que se debe hacer.

En las diferentes aplicaciones se exploran diferentes formas de hacer una aplicación híbrida algunas utilizan react native, otras prefieren el uso de Ionic, en el caso de la investigación de Vilalta se usa la herramienta Flutter, el cual se caracteriza por 3 cosas, velocidad, diseños flexibles y comportamiento nativo. Además, podemos observar la forma en el cual se plasma los prototipos de la aplicación híbrida y se elaboran estrategias para mejorar la interfaz de usuario como también la experiencia de usuario.

4.2.4 Integración de aplicaciones

Con las tecnologías y herramientas antes mencionadas buscamos integrar los sistemas de información de la empresa TI de manera eficiente. Entre los beneficios que pretendemos obtener con la integración de los sistemas de información son:

- Mejora el proceso de la gestión y con ello la toma de decisiones.
- Comunicar eficientemente los diferentes sistemas de información, con la Integración de todas las áreas de una organización
- Salvaguardar los datos.
- Automatizar y simplificar los procesos.
- Garantizar la actualización inmediata de los datos.

Aquello son objetivos que se buscan lograr con la integración de los sistemas, de este modo se produce un mayor valor agregado. Por ello la integración es parte de la estrategia de las empresas.

A continuación, presentaremos investigaciones donde la investigación también apuntó a la integración de sistemas y cómo estos han conseguido los objetivos y mejoras previamente mostradas.

Según Mowla y Kolekar (2020) en su trabajo “Desarrollo e integración de servicios de E-learning utilizando APIs REST”, Usan una APIS REST para conseguir la integración de un servicio E-Learning, a través de la conexión de múltiples módulos especializados, los cuales se alojan en diferentes servidores.

El cliente deberá registrarse y tras eso recibirá un token de autentificación, este token se enviará cada vez que se necesite usar una de las funcionalidades de la

API. Brinda los siguientes servicios, y cada uno se almacena en un servidor distinto.

- Content-as-a-service (servicio del contenido educativo)
- Assignment-as-a-service (servicio para asignación de exámenes y cuestionarios)
- Quiz-as-a-service (servicio de examines)
- Forum-as-aservice. (Servicio de cuestionarios)

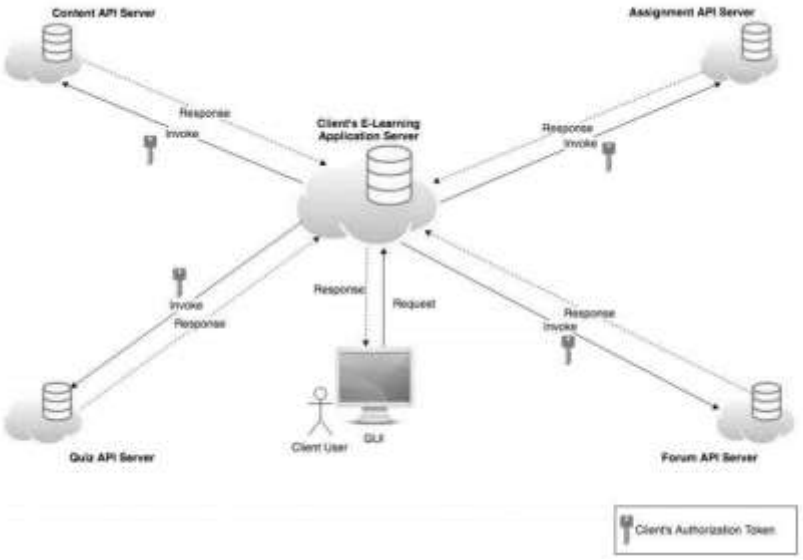


Figura 29: Arquitectura del Servicio E-Learning

Fuente: Mowla y Kolekar

Figura 30: Diagrama de entidad relación

Fuente: Mowla y Kolekar

Con ello el documento, describe de forma específica las características de la integración de los servicios necesarios para conseguir el funcionamiento de una plataforma E-Learning (Mowla & Kolekar, 2020).

De hecho, este trabajo, nos ofrece información detallada sobre los modelos, y consultas de CRUD, de los métodos del REST, esto nos ayudará a guiarnos, con la elaboración de los métodos de la propuesta de nuestra API REST, además el servicio completo se compone de la integración de estos servicios para que en conjunto se pueda dar el examen y evaluación de los alumnos, cada parte es indispensable.

Según Wijayanto, Maryanto, Rahayu, y Iskandar (2019), en su proyecto “The development of REST API-based android application for Micro, Small and Medium Enterprises (MSME) in Purbalingga Regency”, explica sobre la creación de una

aplicación android para las micro, pequeñas y medianas empresas dispersas por toda la basta regencia de Purbalingga. Estas empresas, pueden conectarse al sistema del MYPIME, donde pueden anunciar sus productos, búsqueda de trabajadores, y realizar otras actividades económicas, permitiendo de este modo que la economía mejore para los sectores. Pero existen algunas empresas, alejadas en sectores rurales que no pueden tener acceso

Para solucionarlo, se desarrolla una aplicación móvil basada en APIs REST permitiendo que el sistema sea más accesible para estas empresas. En cada etapa se van logrando las integraciones necesarias para la que la aplicación web sea funcional, se desarrolla un servidor con su respectiva APIs REST, el cual se va a vincular al sistema del MYPIME, para el registro de los empresarios, al servidor Purbalingga's SME para poder hacer uso de las herramientas de negocio, y demás servicios que permitan a los pobladores acceder a las herramientas del MYPIME. La nueva información generada se guardará en una base de datos UMKM (Wijayanto, Maryanto, Rahayu & Iskandar, 2019).

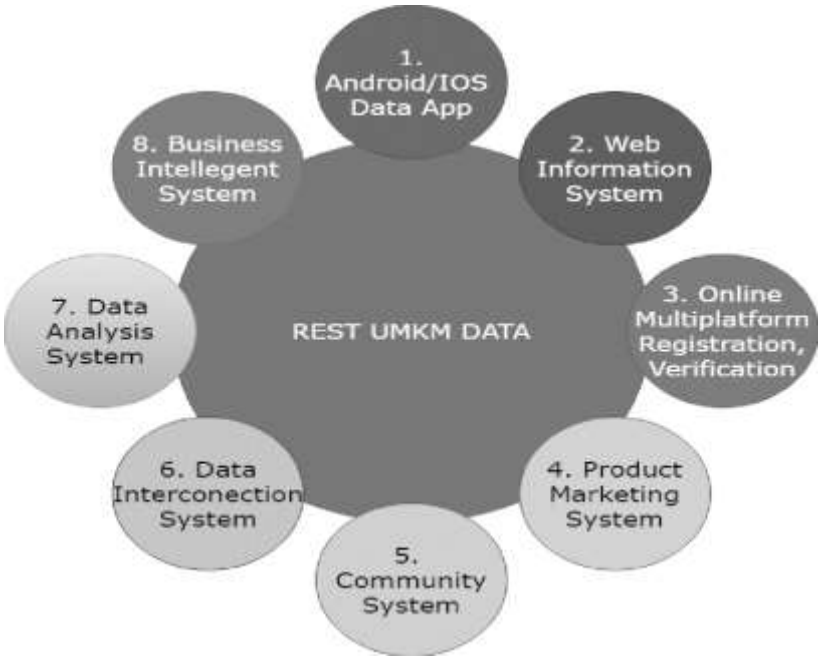


Figura 31: Arquitectura del plan de desarrollo de la aplicación

Fuente: Wijayanto, Maryanto, Rahayu y Iskand

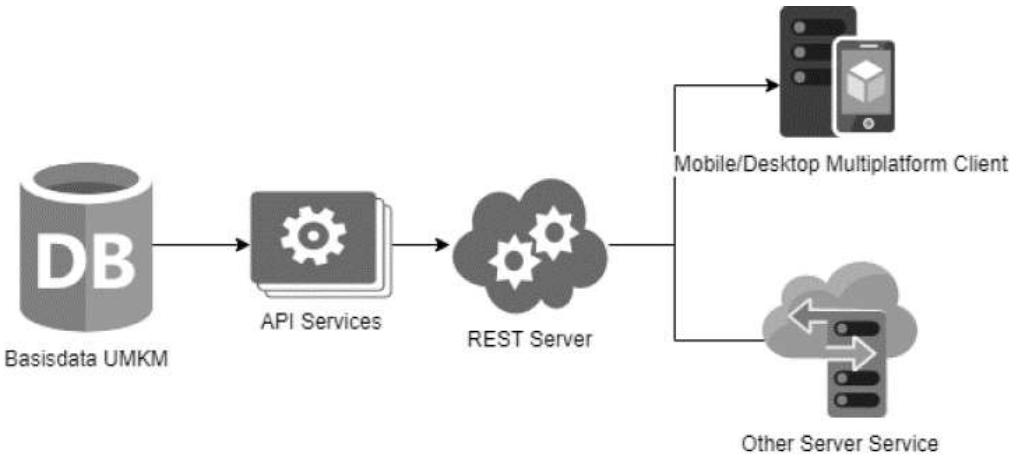


Figura 32: Diseño del sistema

Fuente: Wijayanto, Maryanto, Rahayu y Iskand

Por ello, podemos reforzar la idea de que una API REST es capaz de cumplir la integración de varios servicios en uno solo, permitiendo una mejora en el uso de estos sistemas, en conjunto que, por separado, lo cual fue de gran ayuda para comunidad, de este modo los negocios tuvieron acceso a todas las herramientas del sistema a través del celular, mejorando la accesibilidad al sistema.

Aquí vimos como la integración fue capaz, de crear un nuevo sistema que necesitaba aquellas partes para ser totalmente funcional, y otro servicio que logró aumentar su accesibilidad con esta, ayudando a sus usuarios, aumentando la eficiencia de este sistema, pues además los usuarios no solo tendrán que perder tiempo y hacer un proceso complejo, consultando las herramientas uno por uno, sino que todos están integrados.

Según, André, Beatriz, Filipa, Sandro Catarina y Marisa (2018) en su proyecto de "Colleting information about air quality using smartphones", declaran que la contaminación del aire es un problema grave de salud pública, y que requiere de grandes cantidades de información para poder tratarse adecuadamente, ante ello proponen el concepto de convertir cada teléfono inteligente en un nodo sensor dinámico que se puede utilizar para contribuir activamente a la plataforma centralizada que reúne todos los datos de los nodos sensores.

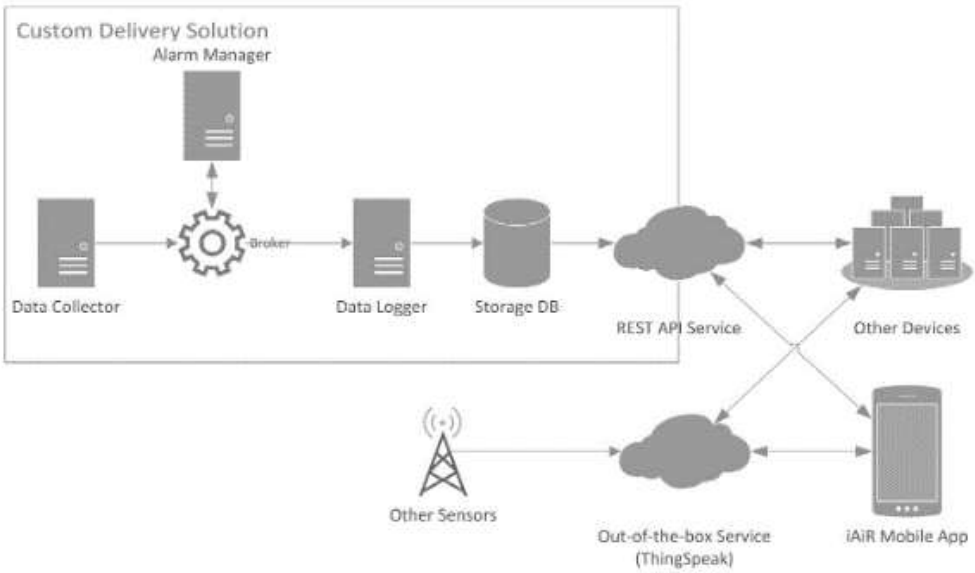


Figura 33: Arquitectura del sistema

Fuente: André, Beatriz, Filipa, Sandro Catarina y Marisa

En su arquitectura, propone que los datos adquiridos por los nodos (celulares) y otros sensores, usen una interfaz RESTful API para que el servicio de análisis de la calidad del aire pueda recibir dichos datos y devolver las consultas de información. Gracias a que la API REst, permite diversas implementaciones de entrega, con solo escribir una pequeña sección de código (André, Beatriz, Filipa, Sandro Catarina & Marisa, 2018).

Aquello nos sirve en nuestro proyecto para justificar la integración entre sistemas, sin importar si son web o de escritorio, con las REST se puede integrar la información que llegue, lo cual ayuda a justificar nuestra propuesta, pues se trata de una aplicación híbrida.

Según Silva, Favarim, Loureiro, Brito, y Todt (2019), en el proyecto “Sensor Monitoring and Supervision Using Web Applications and Rest API” Para el trabajo en zonas industriales, se usa muchas veces diversos sensores que deben ser manejados de forma manual, lo cual permite que puedan existir problemas de confiabilidad en la detección de inconformidades, además de ralentizar la velocidad de la actualización de la información. Por ello se recurre al uso de tecnologías para optimizar e integrar estos procesos, una de estas muy útil es REST, que se basa en gran medida a HTTP, representando objetos mediante un Identificador Uniforme de recursos, y manipulándolos con los métodos GET, POST, PUT, DELETE. Otra característica es su comunicación mediante mensajes con formato JSON y XML, proporcionando una estandarización de servicios y facilidad de integración.

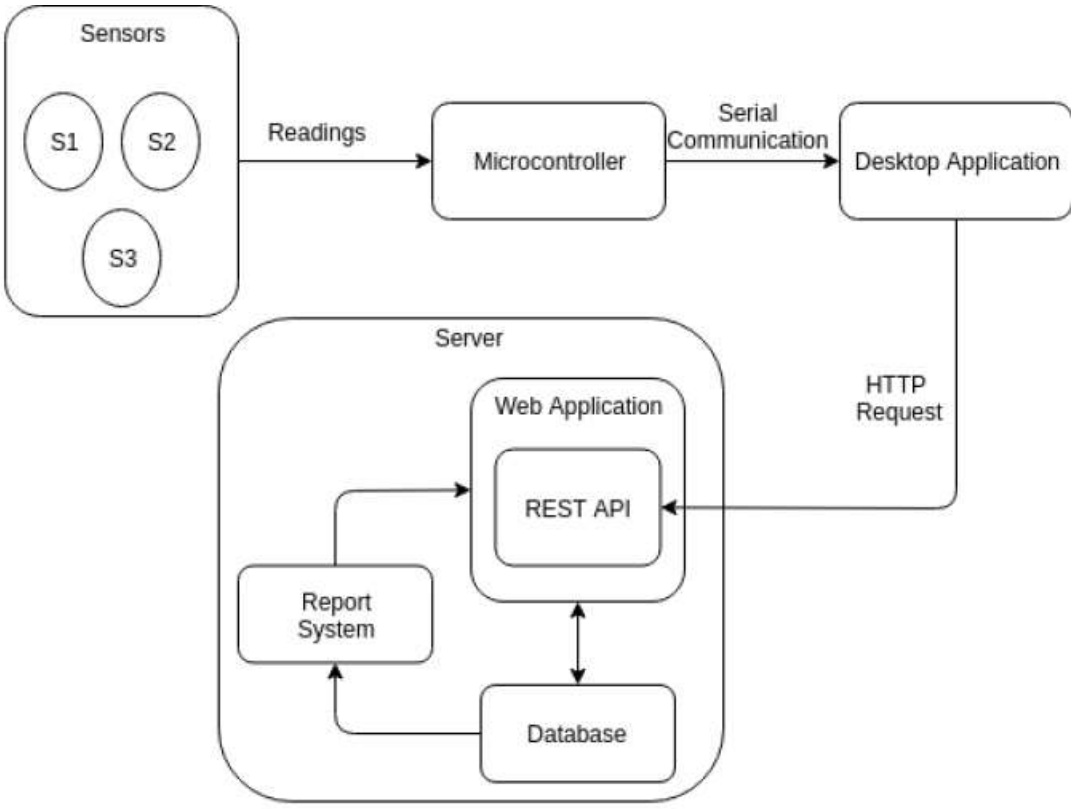


Figura 34: Conexiones del sistema

Fuente: Silva, Favarim, Loureiro, Brito, y Todt

Para la prueba, se usaron sensores conectados a un microcontrolador, el cual lee los datos que envía el sensor y permite enviarlos con formato JSON, por una interfaz de serie a una aplicación de escritorio, que envía la información a la web con el uso de APIS REST, provocando un registro de alarma si la temperatura es muy alta. API REST, que permite realizar el almacenamiento, la configuración y persistencia de los datos, y proporcionar funcionalidades para adquirir lecturas de sensores, registros de datos, alarmas y visualización de datos a través de objetos JSON (Silva, Favarim, Loureiro, Brito, & Todt 2019).

4.2.5 Temática similar, App y almacenes

Finalmente, se toma en cuenta investigaciones que tienen una temática similar a la solución a proponer, en este caso todo referido a aplicaciones de ventas y almacenes.

Según, Perea (2019), en su trabajo para la UTP, “Implementación de un sistema de ventas, producción y almacén para una empresa fabricadora de plástico”. Actualmente las empresas creen que basta con tener un sistema contable y software de compraventa, pero esto en el mercado competitivo no es suficiente. Ante ello toda empresa requiere optimizar constantemente los procesos desarrollados en las diferentes áreas. Con el fin de lograr tener una empresa sana y competitiva, es necesario mejorar la producción tratando de no perder insumos, ir incrementando las ventas y tener actualizado el stock (Perea, 2019).

Además, esta investigación Perea (2019), indica que la empresa necesita un software de gestión y control de los procesos principales, ya que estos se realizan manualmente, consumiendo cantidades altas de recurso humano y financiero, que arrojan resultados erróneos. Por lo tanto, para resolver estos problemas, Perea platea elaborar un software para la gestión y control de las necesidades según el problema crítico y los objetivos de la empresa. Para ello se procede al desarrollo de la solución, se define los requerimientos, se realizan los diferentes diagramas. Además de la creación de prototipos.

Esta investigación nos muestra puntos importantes a tener en cuenta respecto al modelo de aplicaciones que tomaremos como nuestra propuesta de integración. En primer lugar, podemos mencionar de forma general los módulos que tiene una aplicación para almacén y ventas, estos son: registrar las ventas, mostrar el stock y reportes de productos acabado, visualización de las ventas hechas, actualizar precios y las fechas de estos cambios, entre otros requerimientos.

Además, nos amplía el panorama sobre cómo está estructurado el negocio y como lo estará el sistema, como ejemplo tenemos a los actores del negocio y del sistema. En el negocio tenemos como actores a 6 colaboradores. Mientras que en el sistema hay 4 actores. Esto nos muestra los usuarios que influyen en la aplicación.

En segundo lugar, las tablas que el autor plantea para realizar la base de datos. Podemos darnos cuenta de que es un sistema extenso en base a su diagrama de entidad relación mostrado, el software también es extenso ya que aborda almacén, producción y venta en vez de solo uno de esos 3 puntos. Al poder visualizar el detalle de sus diagramas podemos entender mejor como debe realizar el diseño de una aplicación integradora, que pueda consumir los datos de las dos aplicaciones y pueda mostrarlos por la interfaz para ser gestionados y actualizar las dos aplicaciones. Además, que al realizar una app integradora en base a software de almacén y ventas debemos entender todo lo referente al funcionamiento del negocio y sobre las aplicaciones existentes que ayudan a los procesos ya mencionados.

Finalmente, podemos observar todo lo referente a la interfaz de usuario, se muestra unos mockups a grandes rasgos como irán ubicados los botones y los otros elementos UI, aunque a comparación de la investigación de Vilalta nos damos cuenta no se ha tomado en cuenta mucho la experiencia de usuario, pero las interfaces de usuario cumplen con su objetivo. Asimismo, por lo antes mencionado creemos que este trabajo de investigación nos ayudará a dilucidar dudas referentes a: Software para el área ventas, producción y almacén.

Según Gernun (2017), en su trabajo final de máster “Gestión de inventario, stock y almacenes, una alternativa moderna” comenta que la investigación está hecha con el fin de poder brindar una solución tecnológica moderna para la necesidad de

poder gestionar de manera correcta la entrada y salida de todos los productos del almacén (el stock) mediante una interfaz intuitiva y sencilla. Según Gernun, la solución que él brinda contiene mayores módulos funcionales a comparación de otras aplicaciones. Se pretende usar NodeJS para el servidor HTTP y la creación de API Rest, también se usará MySQL para los datos usados por la API. Para las vistas se está utilizando el framework Angular.JS, también se está haciendo uso de la librería Ionic ya que se desea flexibilidad respecto a los diferentes dispositivos en los cuales se cargará la aplicación. Además, se comenta que se usará SQLite para la persistencia de datos en el sistema. Asimismo, se hará uso de Json Web Token para una mejor autenticación. En la investigación se brinda mayor detalle sobre la arquitectura y el diseño del sistema, también sobre los diagramas de entidad relación, entre otros.

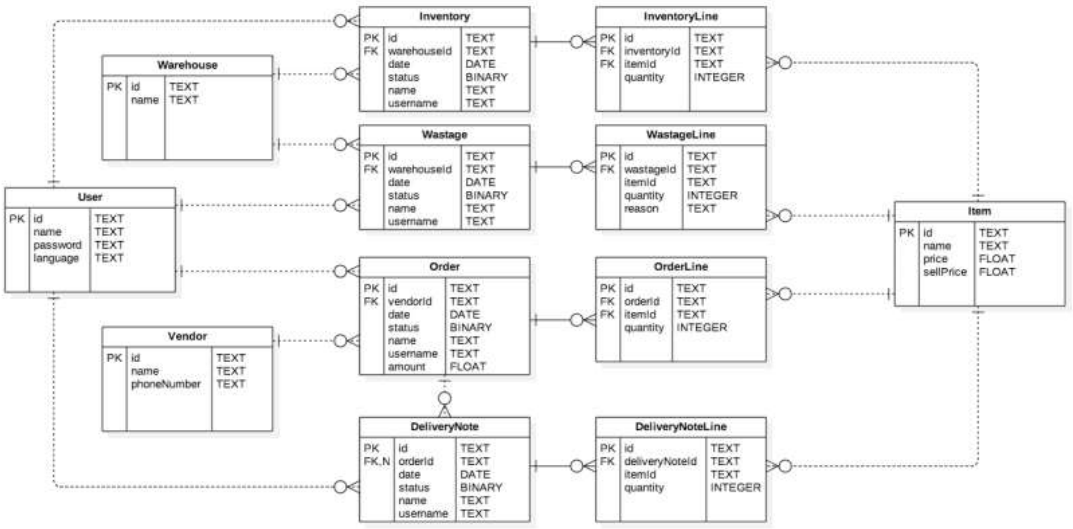


Figura 35: Modelado de los datos

Fuente: Gernun

Estas dos investigaciones nos brindan una perspectiva similar a la temática que tomaremos para nuestra propuesta. La temática es respecto software de almacén y software de ventas, esto ya que se desea proponer una aplicación híbrida que permita integrarlas y se puedan gestionar de una forma más sencilla. La aplicación que propone Gernun se asemeja a nuestra propuesta ya que se hará de forma híbrida con el fin de que sea multiplataforma. La investigación de Perea nos ayuda a saber cómo debemos elaborar de forma detallada el diseño de una aplicación, desde la arquitectura hasta los prototipos de interfaz de usuario. Además, de lo referido a la base de datos como el diagrama de entidad relación.

4.3 RESULTADOS DE LAS FUENTES DE INVESTIGACIÓN

Cuadro de combinación de tecnologías:

En el presente cuadro, comparamos las fuentes exclusivas de SCOOPUS, hechos por autores de renombre, con el conjunto de tecnologías que han usado, para sus respectivos proyectos y trabajo.

Tabla 5: Trabajos y apartado tecnológico

Trabajos y apartado tecnológico	Estructura REST	JSON	JWT	SQL	MÓVIL
Token-based Single Sign-on with JWT as Information System Dashboard for Government.		X	X		X
An analysis on the revoking mechanisms for JSON Web Tokens		X	X		
Microservice Security Agent Based On API Gateway in Edge Computing	X	X	X		
Una API web y desarrollo de aplicaciones WEB, para la difusión de información sobre la calidad del aire	X	X			X
La base de datos eNanoMapper para	X	X			

información de seguridad de nanomaterial					
Analyzing Protein Disorder with IUPred2A	X	X			
Real-Time Human Fall Detection and Emotion Recognition using Embedded Device and Deep Learning	X	X	X		X
Design, simulation and testing of a cloud platform for sharing digital fabrication resources for education	X	X	X		X
Sensor Monitoring and Supervision Using Web Applications and Rest API	X	X			
Mobile Application for Uploading Marks and Student Attendance Management System	X			X	X
The development of REST API-based android application for Micro, Small and Medium Enterprises	X	X			X
A Parallel Computation and Web Visualization Framework for Rapid Large-scale Flood Risk Mapping	X	X			
FuncTree2: An interactive radial tree for functional hierarchies and omics data visualization	X	X			
Colleting information about air quality using smartphones	X	X			X
BRCA-Pathway: Un sistema de integración y visualización estructural de datos de cáncer de mama TCGA en las rutas KEGG	X	X			
Development and Integration of E-learning Services Using REST APIs. International Journal of Emerging Technologies in Learning	X	X	X		X

Fuente: Elaboración propia

Como se puede apreciar, hay una tendencia a usar la estructura REST con JSON, especialmente para el login, además de que muchos autores implementan como

extra el sistema de JWT para más seguridad, aunque algunos se enfocan principalmente en esto. También, se puede observar que muchas de estas son aplicaciones móviles o compatible con aplicaciones móviles, siendo la excepción algunas aplicaciones exclusivas de escritorio, aquellas que tratan de análisis de células (omics), solo por la cantidad enorme de datos que se debe procesar, y memoria a usarse por las herramientas integradas, con lo cual podemos ver que estas tecnologías, siendo este la principal limitante.

Aunque nuestra propuesta de desarrollo no trabajara con información abundante que requieren procesos complejos de conversión de información, tiene como principal función integrar los datos de los sistemas, por lo cual, la hibridación es posible y aceptable.

Tecnologías específicas y justificación:

Hay 2 tecnologías que se usarán para la propuesta, las cuales, a diferencia mencionadas en el otro cuadro, de las demás son menos frecuentes, debido a que no son palabras claves y que, por ende, se podrían optar por otra alternativa para su desarrollo, pero que en si son tecnologías que permitirán cumplir las características que nuestra propuesta necesita, que hemos elegido específicamente por dichas fuentes que las usan y respaldan.

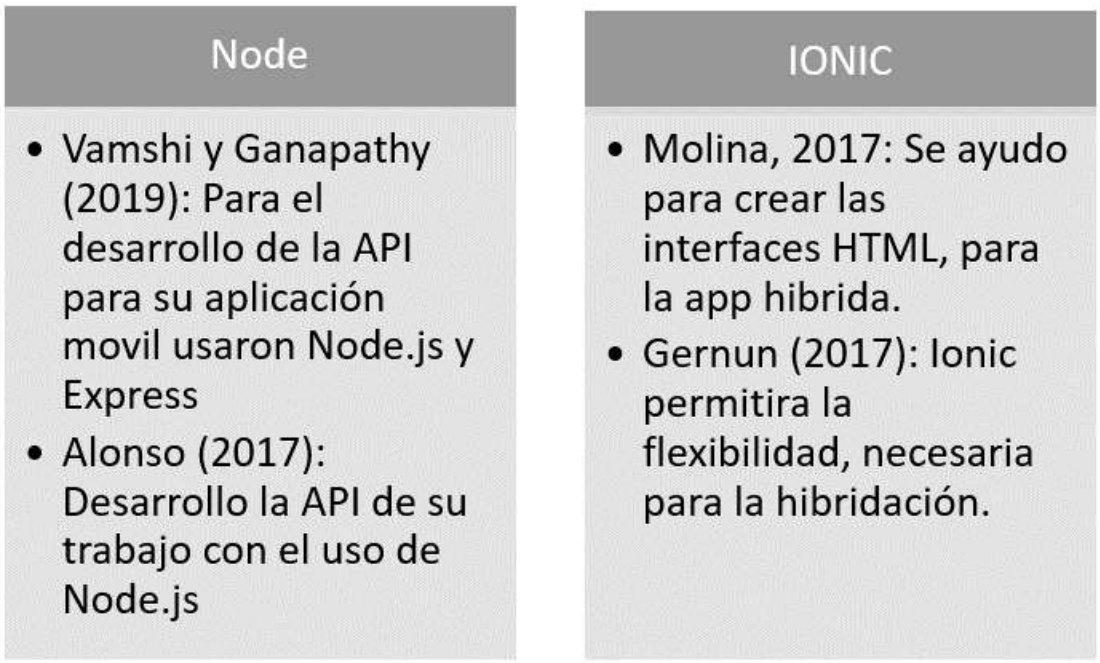


Figura 36: Comparación

Fuente: Elaboración propia

Estas fuentes, justifican el uso de estas tecnologías, en el caso de NODE.JS, este nos permite un desarrollo más sencillo de la API REST, gracias a su librería EXPRESS, que simplifica la elaboración de las consultas REST, razón por la cual hemos agregado esta tecnología, también con la herramienta NPM, se puede traer otras librerías, entre ellas “JSON-SQL-Builder2” que nos permitirá traducir consultas json a sql, aumentando las posibilidades. Para el caso de IONIC, permitiendo la hibridación de la propuesta, de forma más sencilla gracias a las opciones que ofrece para el desarrollo del proyecto.

Comparación REST y GraphQL:

Durante el proyecto, como mecanismo arquitectónico hemos tenido en consideración a si bien nos hemos enfocado en el uso de REST, se debe tener en cuenta la existencia de GraphQL, el cual difiere de rest en la representación de sus datos, y la búsqueda de estos, usando consultas más sencillas y cómodas para el usuario.

En una fuente anterior, de Kleiman y Quintero, titulada “GraphQL vs REST: una comparación de rendimiento a nivel práctico”, se hace una serie de comparaciones sobre las eficiencias de estas, aplicándose en un mismo api con mismos datos y consultas.

Para como métricas se usaron la definición de eficiencia de ISO 9126-3, como el tiempo de respuesta, y el consumo de cache, y según la ISO/IEC 25010, definiendo el throughput como la tasa de transferencia efectiva, el rendimiento.

Tabla 6: Rest versus GraphQL

	REST	**GraphQL**
Promedio de tiempo de respuesta (milisegundos)	11	16
Promedio de uso de cache (bytes)	558 bytes	154 bytes
Promedio del throughput o rendimiento, llamadas respondidas	179	190

Fuente: Elaboración propia

Interpretando estos resultados, se puede decir que Graphql es más eficiente, al aprovechar mejor los datos, y los recursos (memoria caché), pero orientándonos a nuestro caso, se prioriza la eficacia de REST en cuanto la velocidad de transferencia.

Comparación motores de gestión de base de datos:

Muchas investigaciones, usaban estos motores para base de datos, nosotros debemos elegir uno y con esta tabla, que describe las características que tienen en común y diferencias que son relevantes, podemos guiarnos para decidir.

Tabla 7: Sistema de gestión de Base de datos

<table>
<tr><th colspan="3">Sistema de gestión de Base de datos</th></tr>
<tr><td>MySQL</td><td>PostgreSQL</td><td>MongoDB</td></tr>
<tr><td colspan="2">Consultas ordenadas, SQL</td><td>Consultas noSQL</td></tr>
<tr><td>Si se desea empezar un proyecto de negocio, se debe comprar la licencia comercial</td><td colspan="2">Acceso libre y gratuito</td></tr>
<tr><td>Sistema simple</td><td>Un sistema más complejo, para sistemas grandes</td><td>Sistema muy simple, ideal para trabajo con consultas JSON</td></tr>
<tr><td colspan="2">Con la herramienta NPM de Node.js, se puede traer una librería “JSON-SQL-Builder2“que permite traducir consultas JSON a SQL.</td><td>Limitado en cuanto herramientas</td></tr>
</table>

Fuente: Elaboración propia

Las consultas, no sql, no son nuestro objetivo, pues actualmente las consultas ordenadas sql, son las más fáciles de ordenar y analizar cuando se trabaja con dicha información, pues se encuentran normalizadas. Por ende, descartamos MongoDB, y también a PostgreSQL, debido a que la propuesta no se enfoca en un sistema muy

grande, y Mysql nos brinda una interfaz más amigable y sencilla para la gestión. Además, como en el proyecto se usa JWT, se puede usar la librería “JSON-SQL-Builder2” que nos brinda la herramienta NPM de Node.js (como se mencionó antes), para hacer la traducción entre consultas más sencilla.

Propuesta de aplicación que integra aplicaciones de una empresa:

Luego de haber investigado diferentes proyectos que nos hablan sobre la creación de APIs REST que nos ayudan en la comunicación de datos entre el cliente y el servidor, la seguridad de los datos donde podemos mencionar que la autenticación es una parte importante para las aplicaciones y que se deben implementar herramientas como JWT y keycloak, para que este seguro dicho proceso. Además, se investigó sobre la integración de aplicaciones y la creación de aplicaciones híbridas. Por lo que, formulamos una propuesta de solución, en base a todo lo investigado, dicha propuesta es crear una aplicación híbrida que permita integrar aplicaciones informáticas. A continuación, procedemos a mostrar el diseño de la propuesta formulada.

En el proceso de encontrar una solución a la problemática de lograr integrar diferentes sistemas de información de una empresa se pensó en dos propuestas. Por un lado, crear una aplicación nativa, esta permitiría tener acceso a casi la totalidad del uso de los recursos hardware por si algún sistema de una lo requería, pero no sería una aplicación multiplataforma ya que como sabemos se usan diferentes lenguajes de programación para crear aplicaciones IOS y Android como también si se pretende que pueda usarse como web. Por otro lado, la creación de una aplicación hibrida, resultaría beneficioso ya que a comparación de la otra propuesta esta puede usarse para dispositivo móviles como para web en un tiempo menor al que se usaría creando una aplicación nativa.

Para la selección la propuesta de solución se tomaron diferentes criterios. Por un lado, el tiempo que tomaría realizar la aplicación según los puntos clave del tema de investigación como Api Rest, la integración de aplicaciones, entre otros. Por otro lado, las propuestas que tomaremos en cuenta serán las que satisfagan los puntos que debe cumplir la investigación.

Para nuestra propuesta se hará uso de herramientas variadas. Por un lado, herramientas software como “LucidChart” para el diseño de la propuesta, es decir para la creación de diagramas de clase, entidad relación, actividades, secuencias, entre otros. Además, se hará uso de GitHub como repositorio del desarrollo de la aplicación. Asimismo, el uso de “Visual Studio Code” un editor de texto que ayuda al mejor entendimiento de código JavaScript entre otros lenguajes de programación gracias a sus plugins. Asimismo, se utilizará “Node.js” y “Ionic” para la creación de la aplicación. Por otro lado, el uso de hosting que soporten JavaScript para poder probar los avances de nuestro proyecto.

La propuesta se basa en poder integrar diferentes aplicaciones de una empresa privada, por lo que se tomó en cuenta limitarnos a las aplicaciones de venta y de almacenes ya que usualmente estas no se encuentran integradas. El modo usual de funcionamiento en este tipo de empresas es que la aplicación de comercio permita realizar la venta de productos de un establecimiento y llevar la cuenta de su Stock, pero el Stock del almacén principal, es decir no tiene comunicación con esta. Esto hace que poder abastecer de productos a los establecimientos sea tardado. Por lo que nuestra propuesta ayudaría a que esta situación mejore.

Las aplicaciones que deseamos integrar son un software de almacén y un software de ventas. Esto con el fin de que se puede gestionar los elementos (productos), teniendo las dos aplicaciones con los datos actualizados.

La aplicación que se propone utilizará el siguiente patrón de diseño.

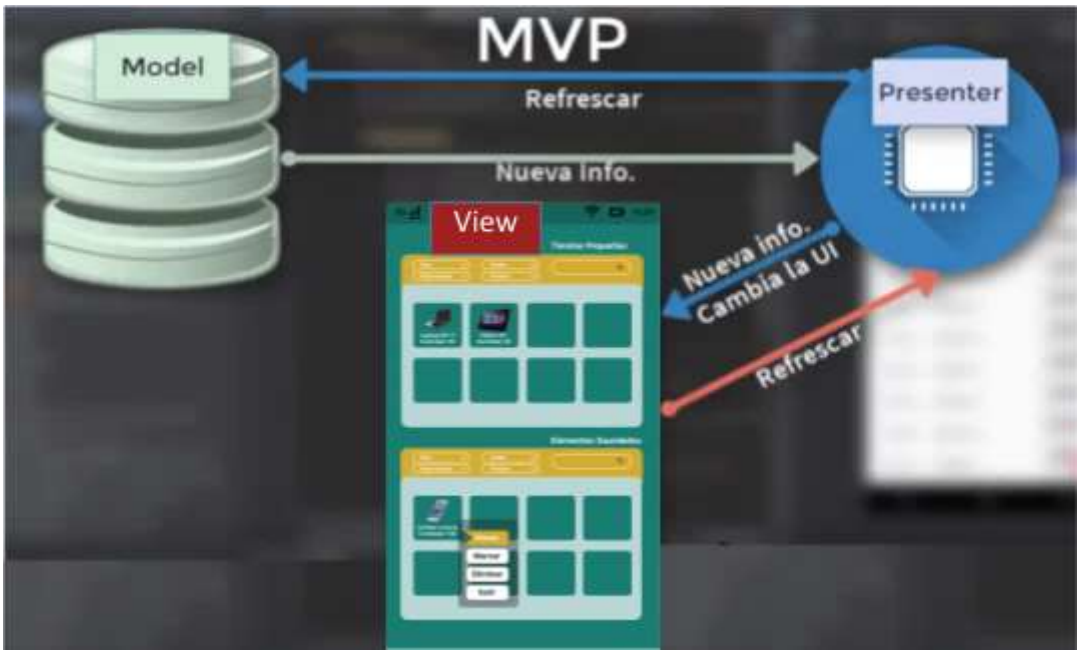


Figura 37: Model View Presenter

Fuente: Elaboración propia

Este patrón es el Model View Presenter, el cual nos ayuda a estructurar un orden en los archivos de código.

Los requisitos de la aplicación son las siguientes:

- Módulo de inicio de sesión de usuarios
- Módulo Almacén de tiendas
- Módulo de Almacén principal
- Módulo de elementos guardados
- Modulo de gestión de alertas

A continuación, se mostrará diferentes diagramas que detallan nuestra propuesta.

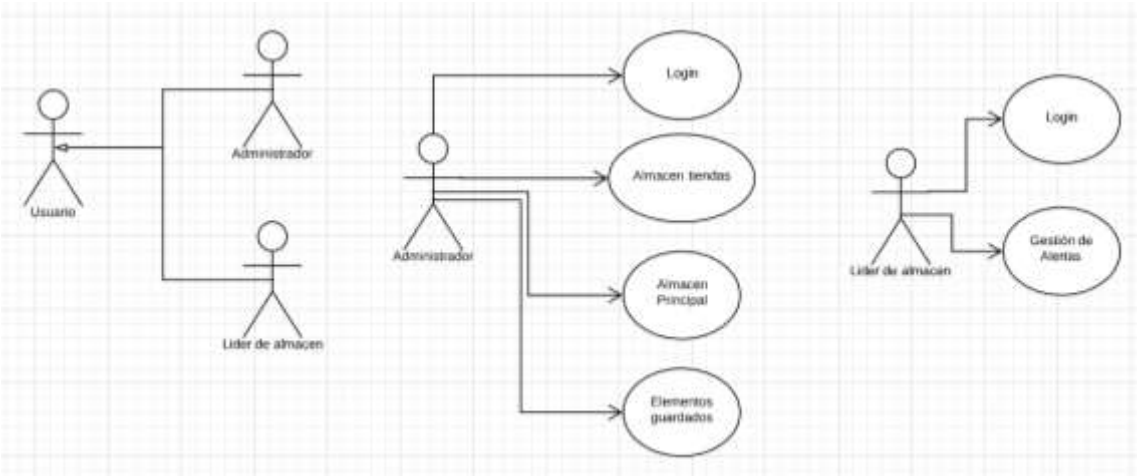


Figura 38: Diagrama de Casos de uso

Fuente: Elaboración propia

Diagramas de actividades:

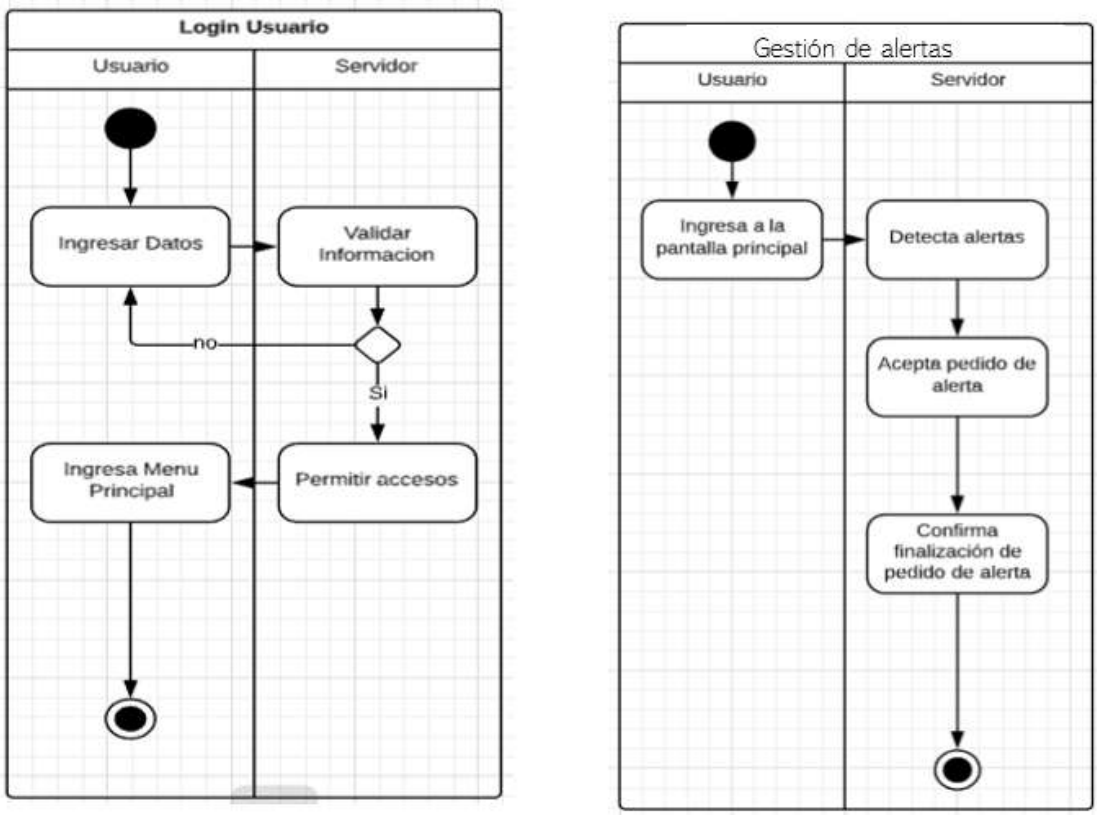


Figura 39: Diagrama de actividades

Fuente: Elaboración propia

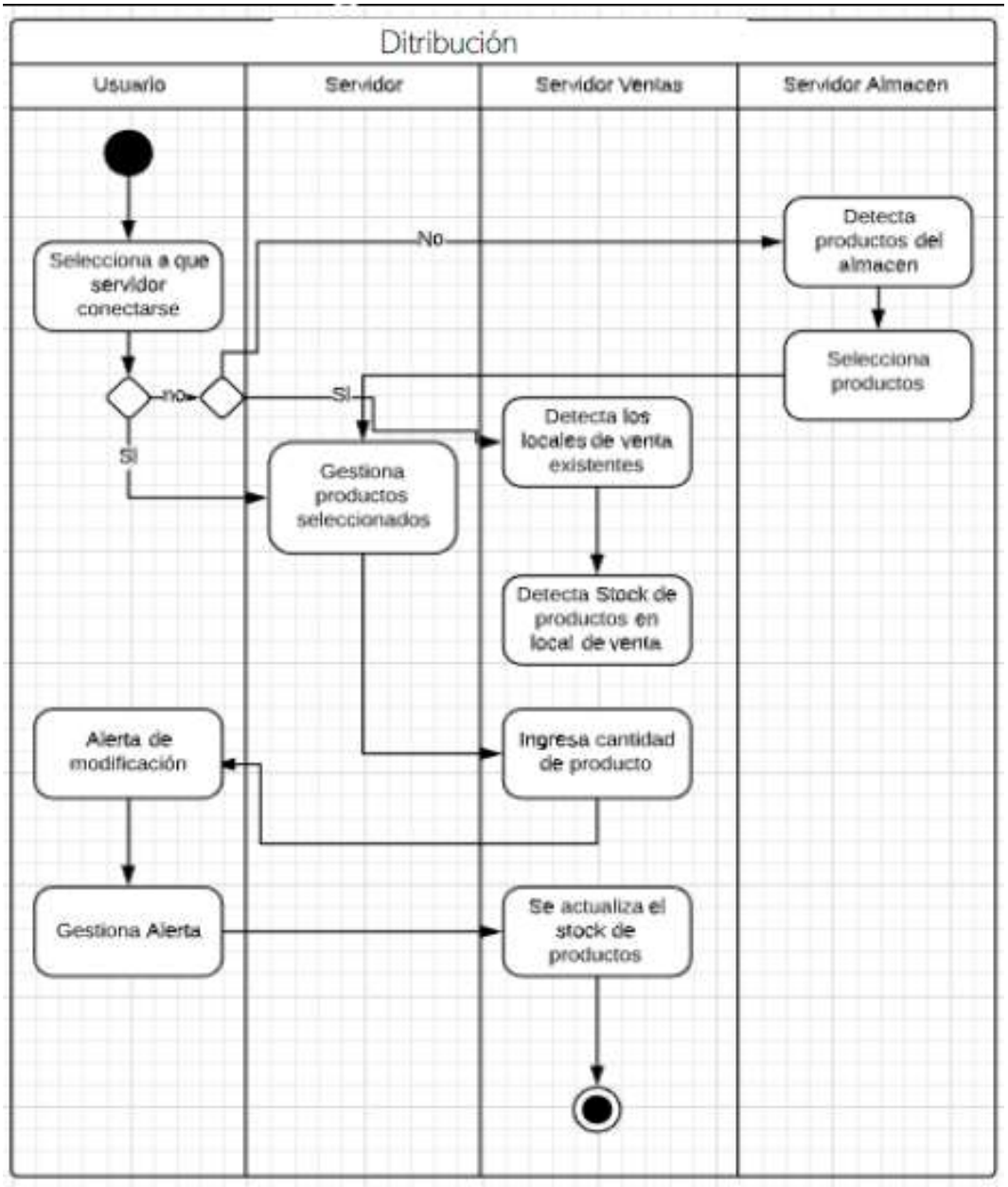


Figura 40: Distribución

Fuente: Elaboración propia

Prototipo: A continuación, se le mostrará un prototipo de la interfaz de usuario hecho con la herramienta “Figma” del software planteado como propuesta de integración de diferentes sistemas informáticos en una empresa.

La primera pantalla que se muestra hace alusión al inicio de sesión del usuario. Esta pantalla está conformada por dos barras de texto donde se debe ingresar el correo y la contraseña, luego hay un botón para iniciar sesión, este último botón nos mandará a la pantalla principal. La segunda pantalla mostrada es la del menú principal, en esta se muestran 3 botones, almacén de tiendas, almacén principal y elementos guardados.

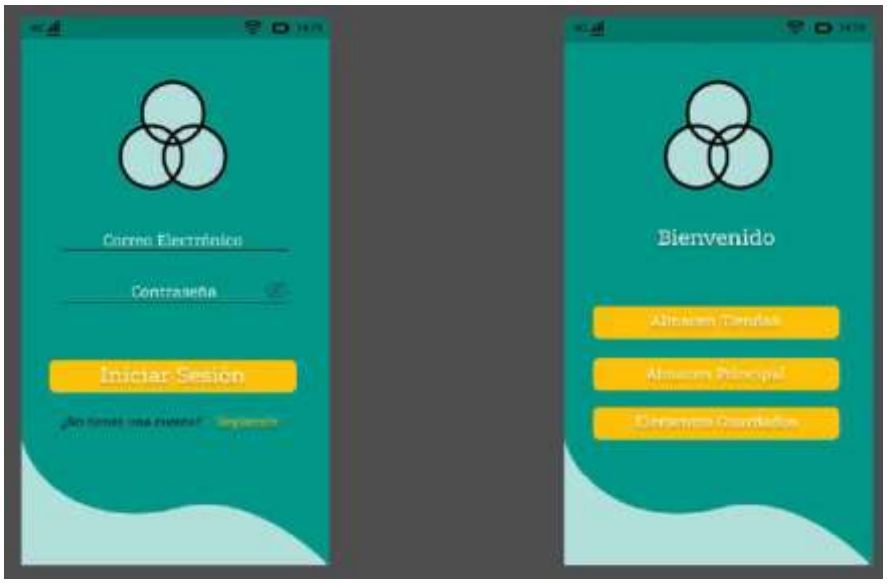


Figura 41: Primera y segunda pantalla

Fuente: Elaboración propia

En la tercera pantalla se muestra la interfaz de almacén de tiendas, se llega a esta pantalla al seleccionar el botón de “almacén de tiendas”. Esta pantalla muestra las diferentes tiendas que hay, mediante los filtros que hay en la parte superior del recuadro podemos buscar la tienda que necesitemos visualizar sus productos. Al seleccionar la tienda, nos mandará a la cuarta pantalla en el cual se visualiza dos recuadros uno donde estarán los elementos en venta de la tienda y en el otro recuadro será para los elementos que queramos almacenar en la tienda,

actualizando así la BD del software de la tienda y alertando al segundo usuario de nuestra aplicación para que envíe los productos actualizados.

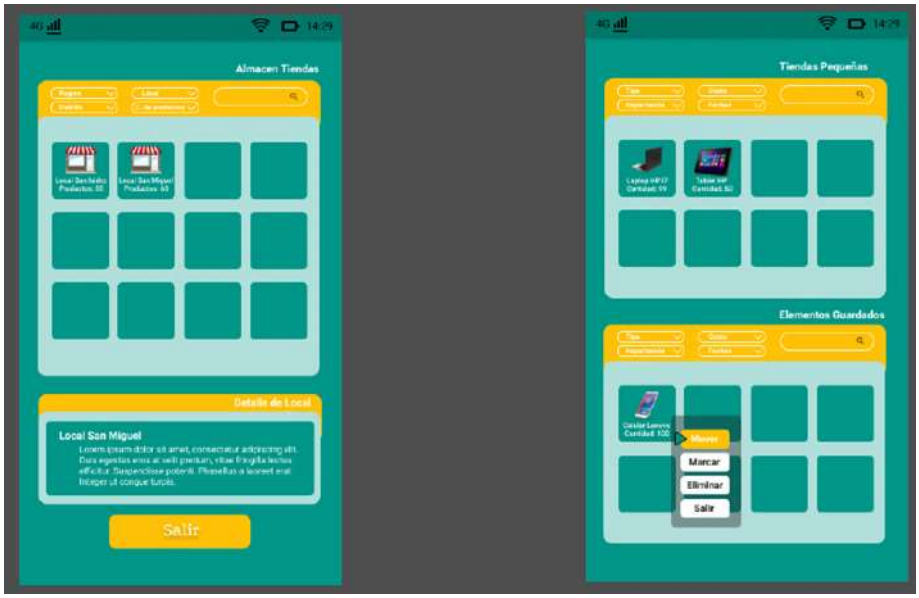


Figura 42: Tercera y cuarta pantalla

Fuente: Elaboración propia

La quinta pantalla nos muestra la sección de “elementos guardados” en el cual podremos visualizar los diferentes que podemos extraer de la pantalla de almacén principal, además tenemos 3 botones, eliminar, marcar, salir. El botón “marcar” es para seleccionar algún elemento que sea importante. En la sexta pantalla se muestra la pantalla de “almacén principal” donde se puede buscar los elementos mediante los filtros y podemos enviarlo a “elementos guardados” para luego actualizar alguna tienda.

Figura 43: Quinta y sexta pantalla

Fuente: Elaboración propia

En la última pantalla se muestra la interfaz principal para el usuario que se encargará de visualizar las alertas y enviar los productos. En esta pantalla mostramos las diferentes alertas y dos botones uno para aceptar que se hará el envío y otro para confirmar que se envió.

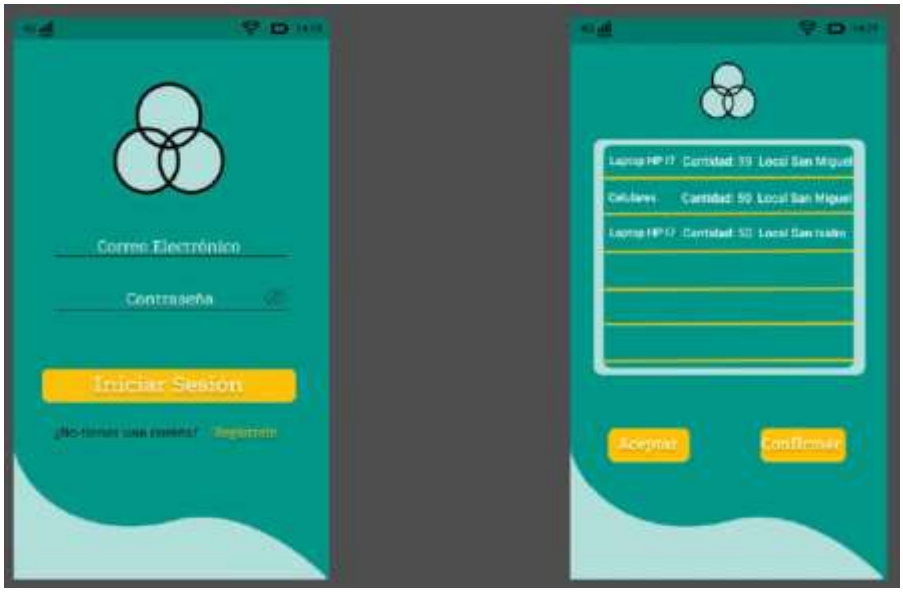


Figura 44: Pantalla final

Fuente: Elaboración propia

CAPÍTULO V: CONCLUSIONES

5.1 TENDENCIAS

Conforme a lo investigado podemos concluir lo siguiente:

Según las diferentes fuentes analizadas hay una tendencia en la integración de aplicaciones, haciendo uso de API para la conexión y comunicación entre estas.

Existe la tendencia de crear APIs usando la arquitectura REST, pues como las fuentes indican, tienen una alta compatibilidad y es relativamente sencillo de implementar.

Más de un autor indica que al crear APIs REST conlleva una serie de vulnerabilidades por lo que optan por el uso de JWT para brindar mayor seguridad de los datos.

Como limitación, podemos decir que en algunos trabajos se comenta que la elección de la arquitectura REST cuando esta se destina al proceso de gran cantidad de datos pesados, el proyecto llega a carecer de rendimiento y

portabilidad. Esto se puede ver en la comparación de GraphQL con REST, esta arquitectura consume mayor cantidad de recursos.

La información recopilada permite establecer una base teórica practica que brinda conocimientos y herramientas básicas, que permite el cumplimiento de los objetivos establecidos en este trabajo de investigación y con ello se da una solución también a las preguntas de investigación propuestas.

Según las investigaciones analizadas, podemos decir que existe la necesidad de realizar integración de aplicaciones para poder mejorar procesos de la empresa.

Integrar aplicaciones puede ser económico y efectivo para las empresas para ello debemos establecer el desarrollo de proyectos software, donde se respeten cada una de sus etapas en su ciclo de vida.

5.2 ENCUENTROS Y DESENCUENTROS ENTRE LOS ESTUDIOS

Como puntos de encuentro podemos indicar lo siguiente:

Más de un autor indica que REST ofrece como característica su simplicidad, por ello los gigantes del internet como Google, Facebook y Amazon desarrolla sus aplicaciones en función de APIs RESTFul.

Según las fuentes revisadas podemos indicar que hay un crecimiento de desarrollo de aplicaciones hibridas por sus diferentes beneficios, esto podemos verlo en la investigación de Gernun como también en la de Molina, entre otros.

Podemos observar que varias empresas desarrollan aplicaciones de forma aislada y que luego tienen la necesidad de realizar una integración de estos. Esto se puede ver en más de una de las investigaciones analizadas.

Como puntos de desencuentro podemos indicar lo siguiente:

Una gran cantidad de autores usan REST para sus aplicaciones, pero Kleiman y Quintero comentan que entre las tecnologías REST y GraphQL, respecto a eficiencia destaca GraphQL pero en cuestión de velocidad de respuesta el mejor es REST.

Según las investigaciones analizadas, podemos decir que gran parte toma en cuenta el uso de JWT para reducir las vulnerabilidades de la autenticación de la aplicación. En cambio, investigaciones como la de Vilalta no toman en cuenta esto y se enfocan en la creación de la aplicación, UI y UX.

Podemos decir que, aunque hay un aumento en el desarrollo de aplicaciones hibridas, la tecnología a usar difiere, esto lo podemos ver en investigaciones como la de Molina, en la de Vilalta y otros.

5.3 RESPONDE A LAS PREGUNTAS DE INVESTIGACIÓN

Respuesta a las preguntas específicas de la investigación:

Esta investigación bibliográfica, tiene como resultado el cumplimiento de los objetivos y con ello se responde las preguntas de investigación planteadas.

¿Cuáles son las funcionalidades que debe tener la aplicación híbrida?

La aplicación, debe ser capaz de manejar los datos de la base de datos del software del almacén y del software de ventas, por ello debe poder conectarse a estas para realizar las siguientes funciones.

Mostrar una lista de productos ubicadas en el almacén principal, los del almacén de cada local de venta y también los seleccionados para estar en “elementos guardados”. También, debe poder permitir que los productos se puedan mover un almacén al otro, y actualizar. Por consiguiente, está la opción de poder agregar y eliminar estos elementos, de esta forma se cumple con todos los métodos del CRUD de una Api REST. Asimismo,

debe tener modulo para la gestión de alertas, el cual debe tener las opciones de “aceptar” y de “confirmar”.

Además, como se indica, debe tener un módulo de inicio de sesión para el usuario, usando JWT para propiciar la seguridad que se requiere, la cual muchas de las fuentes han adoptado aquella herramienta de seguridad para el acceso del usuario a la interfaz.

¿Cómo se da la comunicación entre la aplicación hibrida con las diferentes aplicaciones informáticas de la empresa?

Nuestra propuesta, en cuanto la comunicación, dictamina que, para la conexión se use APIs REST hechas en JavaScript, ejecutadas con NODE.js y usando para su desarrollo la librería Express, que nos brindaran métodos más sencillos para realizar las conexiones a la “Base de datos”, para el manejo de consultas SQL, con los métodos del CRUD, “GET”, “POST”, “UPDATE”, “DELETE”.

De este modo se tendrá acceso a ambos sistemas desde la API y como indica en las fuentes, la hibridación brinda aún más posibilidades de acceso al ser multiplataforma, esto logra una comunicación más amplia.

¿Cómo una aplicación híbrida permite la gestión de datos de la empresa?

Las aplicaciones hibridas tienen la particularidad de poder visualizarse en diferentes plataformas, es decir pueden ser usadas en dispositivos IOS, Android y en computadoras en forma de web. Esta característica ayuda a que los usuarios finales de la aplicación tengan mayor facilidad de acceso a esta y por ende gestionar de mejor forma los datos de la empresa, es decir de forma más rápida y sencilla. Por esta razón en nuestra propuesta se opta por una aplicación hibrida ya que ayudará a que los usuarios que la utilizarán tendrán dichas facilidades y se tendrá una integración de aplicaciones entre almacén y ventas.

¿Cuál es la tecnología que permite establecer una comunicación segura de datos?

Basándonos en las investigaciones analizadas, la tecnología que permite una comunicación de datos segura, respecto a la autenticación del usuario es JWT. Con las funciones que trae Node.js, se puede implementar las consultas con JWT, generando un token para validar la identidad del usuario en el inicio de sesión.

Respuesta a la Pregunta General:

¿La información proporcionada permite el planteamiento de una propuesta de Aplicación Híbrida con APIs REST que integra aplicaciones informáticas de otras plataformas de una empresa?

En función de las respuestas anteriores podemos indicar que se responde la Pregunta general de nuestra investigación bibliográfica. Siendo que si es factible el desarrollo de una aplicación hibrida con una comunicación de datos basada en APIs REST que logran la integración de aplicaciones informáticas de una empresa.

Finalmente, proponemos el desarrollo de una aplicación híbrida, basada en Apis REST, añadiendo también el uso de JSON Web Token (JWT) para proporcionar una seguridad y rápido ingreso al sistema a través de diversas plataformas.

CAPÍTULO VI: REFERENCÍA

6.1 REFERENCIAS BIBLIOGRÁFICAS

Alfonso, M. (2003). Integración de aplicaciones corporativas. [PDF]. Recuperado de http://www.jtech.ua.es/j2ee/2003-2004/modulos/eai/sesion01-traspas.pdf [Consulta: 20 de junio de 2020].

Alonso, S. (2017). API REST y sistema de aprovisionamiento en containers para servIoTicy (Trabajo de final de Grado en Ingeniería Informática de la Universidad de Politécnica de Cataluña y Barcelona Supercomputing Center. España). Recuperado de https://core.ac.uk/download/pdf/87654253.pdf [Consulta: 29 de junio de 2020].

Amodeo, E. (2013). Principios de Diseño de APIs REST. Canada:Leanpub.

Angulo,R. (2013). Aplicaciones móviles híbridas: lo mejor de dos mundos. Revista Debates Instituto de Estudios Superiores de Administración IESA. 18(1), 78-79. Recuperado de http://virtual.iesa.edu.ve/servicios/wordpress/wp-content/uploads/2014/03/e13angulo.pdf [consultado: 23 de junio del 2020]

Arcos, D. (2017). Integración de Oracle y Wordpress para la actualización de contenido en el portal web de la universidad Técnica del Norte. Ibarra, Ecuador. Recuperado

de http://repositorio.utn.edu.ec/bitstream/123456789/7692/1/PG%20578%20TESIS.pdf [Consulta: 19 de junio de 2020].

Arias, P. (2017). Comparación de la Usabilidad de una Aplicación Web una Aplicación Híbrida en Dispositivos Móviles (Trabajo final de investigación del Programa de Estudios de Posgrado de la Universidad de Costa Rica. Costa Rica). Recuperado de http://repositorio.sibdi.ucr.ac.cr:8080/xmlui/handle/123456789/8962 [Consulta: 30 de junio de 2020].

Atencio, D. y Mamani, D. (2017). Interconectividad basada en API REST en aplicaciones de la municipalidad provincial de Lampa. (Tesis de Pregrado). Universidad Nacional del Altiplano. Puno, Perú. Recuperado de http://repositorio.unap.edu.pe/bitstream/handle/UNAP/6163/Atencio_Flores_Dilmerd_Mamani_Machaca_David.pdf?sequence=1&isAllowed=y [Consulta: 19 de junio de 2020].

Bernardo, L. (2010). Proyecto de indagación: La revisión bibliográfica. *Revista de la Facultad de Psicología*. Recuperado de https://www.javeriana.edu.co/prin/sites/default/files/La_revision_bibliografica.mayo_.2010.pdf [Consulta: 16 de junio del 2020]

Bianchini (2018). Desarrollo de un API REST para transmisión de datos de sensores GPS. (Trabajo de Grado). Universidad Politécnica de Valencia. Valencia, España. Recuperado de: https://riunet.upv.es/handle/10251/107770 [Consulta: 19 de junio de 2020].

Buna, S. (2016). Learning GraphQL and relay. London, United Kingdom: Packt Publishing Ltd.

Chanchí, G. E., Campo, W. Y., Amaya, J. P., y Arciniegas, J. L. (2011). Esquema de servicios para Televisión Digital Interactiva, basados en el protocolo REST-JSON. Cuadernos de Informática, 6(1), 233-240.

Cornetta, G., Mateos, J., Touhafi, A., & Muntean, G. M. (2019). Design, simulation and testing of a cloud platform for sharing digital fabrication resources for education. Journal of Cloud Computing, 8(1), 12.

Darzi, Y., Yamate, Y., & Yamada, T. (2019). FuncTree2: an interactive radial tree for functional hierarchies and omics data visualization. Bioinformatics, 35(21), 4519-4521. https://academic.oup.com/bioinformatics/article/35/21/4519/5475599 [Consulta: 20 de junio de 2020].

Erdős, G., & Dosztányi, Z. (2020). Analyzing Protein Disorder with IUPred2A. Current Protocols in Bioinformatics, 70(1), e99. doi:10.1002/cpbi.99

Feria, C. (2017). Seguridad para el control de acceso a recursos de la aplicación web de facturación electrónica Openfact, Ahren contratistas generales - Ayacucho, 2017 (Trabajo final de investigación para optar por el Título Profesional de Ingeniero de Sistemas de la Universidad Nacional de San Cristóbal de Huamanga. Ayacucho. Perú). Recuperado de: http://repositorio.unsch.edu.pe/handle/UNSCH/3376 [Consulta: 30 de junio de 2020].

Fraga, S. (2017). Ampliación de una aplicación para la gestión de presencia en un edificio. (Trabajo de Fin de Grado). Recuperado de http://castor.det.uvigo.es:8080/xmlui/bitstream/handle/123456789/84/TFG_samuel_fraga_mateos.pdf?sequence=1 [Consulta: 22 de junio del 2020]

García, A. (2017). Restful: UN caso de uso de gestión de bibliotecas. (Proyecto de fin de grado). Universidad Politécnica de Madrid. Madrid, España. Recuperada de

http://oa.upm.es/45203/12/TFG_ADOLFO_RODRIGO_GARCIA_NUNEZ.pdf [Consulta: 19 de junio de 2020].

Gernun, J. (2017) Gestión de inventario, stock y almacenes, una alternativa moderna (Trabajo final para Master universitario en Desarrollo de aplicaciones moviles en “Universitat Oberta de Catalunya”). Recuperado de http://openaccess.uoc.edu/webapps/o2/handle/10609/65705 [Consulta: 30 de junio de 2020].

Henry V., Tobias F., Vu, H., Fertig, T., & Braun, P. (2019). Model-Driven Integration Testing of Hypermedia Systems. Recuperado de https://www.riverpublishers.com/journal_read_html_article.php?j=JWE/18/4/5 [Consulta: 20 de junio de 2020].

Kleiman, A., y Quintero, R. (2018). GraphQL vs REST: una comparación de rendimiento a nivel práctico (Trabajo de fin de Grado). Universidad Metropolitano. Venezuela. Recuperado de: https://www.researchgate.net/publication/329155233_GraphQL_vs_REST_una_comparacion_desde_la_perspectiva_de_eficiencia_de_desempeno [Consulta: 17 de junio del 2020]

Martínez Ortuño, N. (2016). Diseño e implementación de una aplicación móvil multiplataforma gestionable por web para la notificación de eventos locales. Recuperado de https://repositorio.upct.es/handle/10317/6131?show=full [Consulta: 12 de junio del 2020]

Molina, A. (2017). Desarrollo de módulos de funcionalidad para aplicación móvil utilizando tecnologías de desarrollo híbrido para la empresa BSD Enterprise S.A de C.V. (Reporte Final de Estadía). Recuperado de

http://reini.utcv.edu.mx/bitstream/123456789/241/1/007162.pdf [Consulta: 23 de junio del 2020].

Mowla, S., & Kolekar, S. V. (2020). Development and Integration of E-learning Services Using REST APIs. International Journal of Emerging Technologies in Learning (iJET), 15(04), 53-72.

Muñoz, F. (2019). Diseño e implementación de una arquitectura full stack con software gratuito. (Doctoral dissertation). Universidad Politécnica de Valencia. España. Recuperado de https://riunet.upv.es/bitstream/handle/10251/125982/Mu%C3%B1oz%20-%20Dise%C3%B1o%20e%20implementaci%C3%B3n%20de%20una%20arquitectura%20full%20stack%20con%20software%20gratuito.pdf?sequence=1 pdf [Consulta: 20 de junio del 2020].

Muyon, C., y Montaluisa, F. (2020). Métodos de seguridad de la información para proteger la comunicación y los datos de servicios web REST en peticiones HTTP utilizando JSON Web Token y Keycloak Red Hat Single Sign On. Iberica, E29, 198-213. Recuperado de https://search.proquest.com/openview/bb69452d485a366de28de7b3110c4461/1?pq-origsite=gscholar&cbl=1006393 [Consulta: 19 de junio de 2020].

Navarro, M. (4 de diciembre de 2017). Plataformas para la integración. *Byte*. Recuperado de https://revistabyte.es/tema-de-portada-byte-ti/plataformas-la-integracion/ [Consulta: 15 de junio de 2020].

Nebel,A. (2018). Arquitectura de Microservicios para Plataformas de Integración. (Tesis de Maestría). Universidad de la República. Montevideo. Uruguay. Recuperada de https://www.colibri.udelar.edu.uy/jspui/bitstream/20.500.12008/20586/1/tm-nebel.pdf [Consulta: 18 de junio de 2020].

Ñique, V. (2016). Implementación de solución de autenticación segura basada en doble factor en una entidad del estado. (Tesis de Pregrado). Universidad San Ignacio de Loyola. Lima, Perú. Recuperada de http://repositorio.usil.edu.pe/bitstream/USIL/2481/1/2016_%C3%91ique_Implementacion_de_solucion_de_autenticacion.pdf [Consulta: 10de junio de 2020].

Ortega, J. (2017). Aplicación Android y Servicios REST para Compartir Libros Electrónicos. (Trabajo fin de Master). Universidad de Alicante, Alicante, España. Recuperada de https://rua.ua.es/dspace/bitstream/10045/68129/1/Aplicacion_Android_y_servicios_REST_para_compartir_lib_Ortega_Bastida_Javier.pdf [Consulta: 19 de junio de 2020].

Parraguez, S., Chunga, G., Flores, M., & Romero, R. (2017). El estudio y la investigación documental: Estrategias metodológicas y herramientas TIC. Chiclayo: EMDECOSEGE. Recuperado de https://www.academia.edu/32220610/El_estudio_y_la_investigaci%C3%B3n_documental_estrategias_metodol%C3%B3gicas_y_herramientas_TIC [Consulta: 17 de junio del 2020]

Perea, V. (2019). Implementación de un sistema de ventas, producción y almacén para una empresa fabricadora de plástico. (Tesis de pregrado). Universidad Tecnológica del Perú. Lima. Perú. Recuperada de http://repositorio.utp.edu.pe/bitstream/UTP/1901/1/Victor%20Perea%20Pereyra_Trabajo%20de%20Suficiencia%20Profesional_Titulo%20Profesional_2019.pdf [Consulta: 19 de junio de 2020].

Pereira, A., Patrício, B., Fonte, F., Marques, S., Reis, C. I., & Maximiano, M. (2018). Collecting information about air quality using smartphones. Procedia computer

science, 138, 33-40. https://www.sciencedirect.com/science/article/pii/S1877050918316375?via%3Dihub [Consulta: 15 de junio de 2020].

Pratama, A.,Linawati,L., & Putra, N. (2018). Token-based Single Sign-on with JWT as Information System Dashboard for Government. Revista Telkomnika. 16(4), 1745-1751. Recuperado de http://journal.uad.ac.id/index.php/TELKOMNIKA/article/view/8388 [Consulta: 22 de junio del 2020]

Puerta, G. (2015). Desarrollo de una API para la descripción y gestión de Servicios Web REST. (Trabajo de fin de Máster). Universidad Jaume I. España. Recuperado de http://repositori.uji.es/xmlui/handle/10234/156006 [Consulta: 12 de junio de 2020].

Santos, W. y Serrano, J. (2017) Desarrollo de una API REST con sus aplicaciones web y móvil para la venta de ropa online de la empresa Rosman. (Tesis de Pregrado). Universidad central del Ecuador. Recuperado de http://www.dspace.uce.edu.ec/bitstream/25000/9668/1/T-UCE-0011-312.pdf [Consulta: 19 de junio de 2020].

Silva, A. L., Favarim, F., Loureiro, J. F., Brito, R. C., & Todt, E. (2019, October). Sensor Monitoring and Supervision Using Web Applications and Rest API. In Proceedings of the Intelligent Embedded Systems Architectures and Applications Workshop 2019. *INTESA.* 29-33. doi.org/10.1145/3372394.3372398

Sreenidhi, I., Satyanarayana, P. (2020) Real-Time Human Fall Detection and Emotion Recognition using Embedded Device and Deep Learning. International Journal of Emerging Trends in Engineering Research 8(28), 780-786. doi.org/10.30534/ijeter/2020/28832020

Vamshi, A., Ganapathy, K. (2019) Mobile Application for Uploading Marks and Student Attendance Management System. International Journal of Engineering and Advanced Technology, 8, 379-383. Recuperado de https://www.ijeat.org/wp-content/uploads/papers/v8i6S/F10800886S19.pdf [Consulta: 30 de junio de 2020].

Vidalta, T. (2019). Desarrollo de Partyfy, una app híbrida en React Native (Trabajo final para el grado de ingeniera informática en la Escuela Politécnica Superior). Recuperado de https://repositori.udl.cat/bitstream/handle/10459.1/67973/tvilaltav.pdf?sequence=1&isAllowed=y [Consulta: 30 de junio de 2020].

Viktor, L., Levendovszky, J., & Ekler, P. (2018). An analysis on the revoking mechanisms for JSON Web Tokens. Revista Next Generation Internet of Thigs (IoT) and Cloud Security Solutions. 14(9). Recuperado de https://journals.sagepub.com/doi/pdf/10.1177/1550147718801535 [Consulta: 22 de junio del 2020]

Wang, M., Lai, R., Xia, R., Wang, M., & Yang, M. (2019, August). A Parallel Computation and Web Visualization Framework for Rapid Large-scale Flood Risk Mapping. In Journal of Physics: Conference Series. 1288. doi:10.1088/1742-6596/1288/1/012065

Wijayanto, B., Maryanto, E., Rahayu, S. P., & Iskandar, D. (2019, November). The development of REST API-based android application for Micro, Small and Medium Enterprises (MSME) in Purbalingga Regency. In Journal of Physics: Conference Series. 1367. doi:10.1088/1742-6596/1367/1/012005

Xu, R., Jin, W., & Kim, D. (2019). Microservice Security Agent Based On API Gateway in Edge Computing. Sensors, 19(22), 4905. doi: 10.3390/s19224905

6.2 ANEXOS

6.2.1. GLOSARIO DE TÉRMINOS

API: La API o interfaz de programación de aplicaciones es una serie de especificaciones que permite que la aplicación y el servidor se comuniquen entre sí.

REST: REST o “transferencia de estado representacional” es un estilo de arquitectura que nos permite la comunicación entre cliente y servidor. Actualmente las API creadas paras las aplicaciones utilizan REST.

JSON: Las siglas significan “notación de objetos JavaScript” y es un formato de texto que permite la transferencia de datos, este formato es fácil de interpretar para la computadora y también de generar archivos con dicho formato.

Aplicación hibrida: Es un tipo de aplicación multiplataforma, es decir puede usarse en diferentes dispositivos móviles y como web. Son módulos web dentro de un contenedor nativo. Para realizar este tipo de aplicaciones se usa HTML, CSS, JavaScript.

Framework: Su significado es “marco de trabajo” y es una serie de herramientas y estilos acorde a una estructura base que ayuda al desarrollo de software. Existen muchos framework para los diferentes lenguajes de programación, algunos de estos son Angular, Laravel, entre otros.

UX: Estas siglas se refieren a “experiencia de usuario” y es la usabilidad que tiene el software desarrollado desde el punto de vista del usuario.

UI: Estas siglas se refieren a “interfaz de usuario” y es lo que se visualiza en una aplicación para poder interactuar con este mismo. Esta se puede desarrollar con tecnologías como HTML, CSS, JQUERY, entre muchas otras.